In-operation learning of optimal wind farm operation strategy

Department of Wind Energy Master Report

Joan Oliva Gratacos

DTU Wind Energy-M-0165

June 2017

DTU Wind Energy
Department of Wind Energy

Authors: Joan Oliva Gratacos

Title: In-operation learning of optimal wind farm operation strategy

DTU Wind Energy-M-0165
June 2017

Project Period:
January – June 2017

ECTS: 30

Education: Master of Science

Supervisors:

Nicolaos Antonio Cutululis
Jonas Kazda
DTU Wind Energy

Remarks:
This report is submitted as partial fulfillment of the requirements for graduation in the above education at the Technical University of Denmark.

DTU Wind Energy is a department of the Technical University of Denmark with a unique integration of research, education, innovation and public/private sector consulting in the field of wind energy. Our activities develop new opportunities and technology for the global and Danish exploitation of wind energy. Research focuses on key technical-scientific fields, which are central for the development, innovation and use of wind energy and provides the basis for advanced education at the education.

We have more than 240 staff members of which approximately 60 are PhD students. Research is conducted within nine research programmes organized into three main topics: Wind energy systems, Wind turbine technology and Basics for wind energy.

Technical University of Denmark
Department of Wind Energy
Frederiksborgvej 399
4000 Roskilde
Denmark

www.vindenergi.dtu.dk

Abstract

In a wind farm, power losses due to wind turbine wake effects can be up to 30-40% under certain conditions. As the global installed wind power capacity increases, the mitigation of wake effects in wind farms is gaining more importance. Following a conventional control strategy, each individual turbine maximizes its own power production without taking into consideration its effects on the performance of downstream turbines. Therefore, this control scheme results in operation conditions that yield suboptimal power production.

In order to increase the overall wind farm power production, a cooperative control strategy can be used, which coordinates the control actions among the wind turbines in the wind farm. This work further investigates the model-free Bayesian Ascent optimization algorithm using SimWindFarm and a standalone Dynamic Wake Meandering model-based simulation tool

An advantage of such optimization approach is that the control strategy adapts to operational conditions in the wind farm and is not model-dependent. An approximation of the wind farm power function is constructed using GP regression to fit the control action inputs and the noisy measured power outputs, which is then maximized to determine the optimal control inputs. This estimation is updated in every iteration, allowing the control system to learn from the target system while performing the optimization. The usage of all historical data, along with a trust region constraint in the sampling of new inputs, contribute to a fast convergence rate with gradual changes of the control actions.

The developed learning technique is implemented in a wind farm controller and tested in both SimWindFarm and standalone Dynamic Wake Meandering model-based simulation tools. With the conducted tests, performance of the algorithm is assessed considering the

different dynamics in the wind farm, thus obtaining an accurate representation of real farm operation. The developed controller reliably improves farm efficiency, even with uncertainty present in measurements.

Compared to traditional control strategies, an increase in total wind farm power production is obtained when using a cooperative control strategy. Such enhancement in wind farm performance would result in an improvement of wind farm economics and hence in further growth of wind-energy based power generation.

Acknowledgments

First and foremost, I would like to express my sincere gratitude to my supervisors Jonas Kazda and Nicolaos Antonio Cutululis. More precisely, I would like to thank Jonas for his guidance and continued feedback, and Nicolaos for his valuable contributions. It has been a pleasure to work with both of them.

Secondly, I would like to thank the colleagues I met in DTU Risø for making any working day in the campus that much better.

I also want to thank Ashleigh, Corrado, Davide, Guille, Michele, and Will for all the great experiences we shared in Denmark.

Finally, I would like to thank my family for their continued and unconditional support from a distance.

This work is part of the CONCERT project, which is funded by Energinet.dk under the Public Service Obligation scheme (ForskEL 12396).

Table of Contents

List of Figures

List of Tables

Nomenclature

Symbols

a	Axial induction factor	[-]
A_r	Rotor area	[m^2]
A_w	Wake area	[m^2]
C_P	Power coefficient	[-]
C_T	Thrust coefficient	[-]
D	Rotor diameter	[m]
d	Turbine spacing	[m]
$DerFact$	Derating Factor	[-]
E	Extracted energy	[J]
f	Optimization objective function	[-]
$\dot{m}$	Air mass flow	[kg/s]
η	Conversion efficiency	[-]
N	Number of turbines	[N]
ρ	Air density	[kg/m^3]
P_{farm}	Wind farm power output	[W]
P_i	Turbine power output	[W]
P_r	Turbine rated power output	[W]
P_w	Freestream wind power	[W]

P_∞	Atmospheric pressure	[Pa]
P_r^+	Pressure before turbine rotor	[Pa]
P_r^-	Pressure after turbine rotor	[Pa]
σ_e	Measurement uncertainty	[-]
θ^w	Wind direction	[º]
o	Yaw angle	[º]
T	Rotor thrust force	[N]
τ	Trust region size	[-]
u_c	Cut-in wind speed	[m/s]
u_f	Cut-out wind speed	[]
u_d	Rotor wind speed	[m/s]
u_r	Rated wind spedd	[m/s]
u_w	Wake wind speed	[m/s]
u_∞	Freestream wind speed	[m/s]
X	Control action	[-]
Y	System output	[-]

Abbreviations

BA	Bayesian Ascent
BO	Bayesian Optimization
DADSA	Decentralized Aggressive Discrete Stochastic Approximation
DOE	Design of Experiments
DRAF	Decentralized Regret-Based Adaptive Filtering
EI	Expected Improvement
GA-MPPT	Gradient-Ascent MPPT
GP	Gaussian Process

LS	Least-Squares
MPPT	Maximum Power-Point Tracking
PDLPO	Pay-off Distributed Learning for Pareto Optimality
PSO	Particle Swarn Optimization
QN-MPPT	Quasi-Newton MPPT
sDWM	Standalone Dynamic Wake Meandering
SED	Safe Experimentation Dynamics
SWF	SimWindFarm

Chapter 1

Introduction

As the world transitions towards a sustainable development path, renewable energy sources are gaining more importance. For instance, the European Union has set a goal of 20% total energy consumption coming from renewable sources by 2020. Among the existing renewable energy technologies, wind energy has shown one of the highest potential for large-scale generation. Currently, wind energy is the leading renewable power source in Europe, the United States and Canada. Moreover, worldwide installed capacity reached 433 GW in 2015, which represented a 17% growth over the previous year [1]. However, in order to meet renewable generation goals, further growth of wind-based generation is required.

1.1. Motivation

The rising demand for wind power generation has led to an increase in the size of wind farms, seeking a reduction in the cost of energy. However, a drawback from large turbine arrays is the decrease in farm efficiency due to wake interference. Turbines operating in the wake of other wind turbines experience a reduction in incoming wind speed and an increase in turbulence intensity, which cause a decrease in power production. In large off-shore wind farms, power losses due to wake effects have been reported to be as high as 40 %[2]. Nowadays, conventional wind farm control approaches seek to maximize each individual turbine power production, disregarding the aerodynamic interactions of wind turbines. Consequently, this strategy results in a suboptimal power production.

Therefore, strategies that mitigate wake interference can increase the power production of a wind farm. One possible approach is the implementation of a cooperative control strategy, which adjusts the operating set-point of each individual turbines in a way that losses due to aerodynamic interactions are minimized. Wind farm flow models can be used to derive an analytical wind farm power function, which then can be employed to determine the optimal operation conditions. However, due to the approximation present in the modelling procedure, the optimized solution might not be the true optimum for the real target wind farm. Furthermore, in order to accurately estimate the wind farm power production, these models need to be calibrated for a specific location with on-site data.

In order to avoid the use of an analytical wind farm power function, a data-driven optimization technique can be used. With such approach, power production is maximized using only control action inputs, i.e. turbine operation set-points, and turbine power measurements. Consequently, the optimization procedure is deemed model-free, since it requires no knowledge of the actual wind farm power function.

Several data-driven wind power production optimization algorithms have been proposed in literature [3]–[7]. Among the available procedures, this thesis further investigates the Bayesian Ascent (BA) algorithm. Said technique has been tested in different farm layouts, using an analytical wind power function derived from a continuous wake model based on the Jensen model. Furthermore, experimental tests in a wind tunnel have also been performed. In these tests, the algorithm managed to improve wind farm efficiency in a low number of iterations, while being able to cope with uncertainty in the measurements. The optimization decision variables are the turbine induction and yaw misalignment.

1.2. Objective

The main objective of this thesis is to develop an in-operation learning technique based on the BA algorithm that can improve wind farm power production. Said technique has to be applicable to real time control and offer robust operation under time-varying conditions.

First of all, the optimization algorithm is tested in the maximization of analytical functions. With these tests, the incidence of the algorithm parameters in the optimization is investigated. Additionally, possible improvements to the original formulation of the algorithm are explored.

Finally, the developed in-operation learning technique is tested in induction-based maximization of wind farm power production, using SimWindFarm (SWF) and a standalone Dynamic Wake Meandering (sDWM) model-based simulation tool. The combination of conducted tests aims to simulate the optimization in conditions similar to real wind farm operation.

1.3. Thesis Structure

The presented thesis is organized in nine chapters. The contents of each chapter can be summarized as:

- Chapter 2 presents a basic introduction of wind farm operation, including wind turbine aerodynamics, wind farm aerodynamics, and wind farm control.
- Chapter 3 describes eight wind farm optimization techniques available in literature. Advantages and disadvantages of each technique are discussed. Finally, the reasoning behind the choice of the BA algorithm is presented.
- Chapter 4 discusses the proposed implementation of the BA algorithm. First of all, the wind farm control problem is defined. Then the mathematical formulation of the BA algorithm is given. Finally, a strategy for dealing with time-varying conditions is suggested.
- Chapter 5 shows the results the analytical functions optimization tests. The effects of the algorithm parameters on the optimization are discussed.
- Chapter 6 presents the design process of the trust region approach used in the algorithm. First of all, a discussion on the adequacy of a trust region approach for the wind farm power maximization problem is presented. Then, different trust region approaches are formulated. These alternative approaches are tested in the maximization of analytical functions.
- Chapter 7 presents the tests performed using SWF. A description of the SWF simulation environment is given, and the set-up of the simulation is presented. Finally, the results of the simulations are shown.
- Chapter 8 describes the tests performed using the sDWM model-based simulation tool. First of all, a description of this simulation tool is given. Then the simulation

settings and the BA algorithm configuration are presented. Finally, the results of the simulations are shown.

- Chapter 9 presents the conclusions derived from the thesis and discusses possible future research directions.

Chapter 2

Wind Farm Operation and Control

In this chapter the basics of wind farm aerodynamics and control are presented. First of all, the actuator disk theory for deriving wind turbine power production is formulated. Then an overview on wind turbine aerodynamics and wake effects is given. Finally, the generalities of wind farm control and strategies for wind farm power maximization are presented.

The growth of wind energy during the last decades has resulted in an increase of the number of turbines being installed in wind farms. The grouping of wind turbines generates two issues that need to be addressed: a wind turbine operating in the wake of another turbine has a reduced power production (due to the reduction in incident wind speed) and is subject to more demanding fatigue loads (as a result of an increase in turbulence intensity). As a result of these challenges, traditional wind farm control solutions need to be updated in order to optimize wind farm operation.

2.1. Aerodynamics of Wind Turbines

Wind turbines extract energy from the wind; therefore, a fluid element that passes through the rotor loses part of its kinetic energy. Wind stream conditions, such as speed, direction, and turbulence intensity, play a key role in determining the efficiency of the energy extraction.

The conditions of the upstream wind flow are defined, among others, by the atmospheric values of speed u_∞ and pressure P_∞. As the flow passes through the turbine, it gradually

slows due to the extraction of kinetic energy. Static pressure suffers an increase just in front of the rotor, and then drops suddenly as a result of the force exerted by the blades. The pressure increases gradually as the wind flow travels downstream, tending to the atmospheric value. On the other hand, wind speed needs more time to recover its original value, potentially hindering the performance of downstream turbines. Such phenomena is known as wake effect.

The amount of power that can be extracted from a wind flow is defined by its speed u_∞, the air density ρ and the area swept by the rotor A_r:

$$P_w = \frac{1}{2}\rho A_r u_\infty^3 \tag{2.1}$$

The power output from a single turbine can be modelled using the actuator disk theory, which portraits the wind turbine rotor as a disk, and quantifies its power production as the energy extracted by the disk. Consequently, the analytical expression for wind turbine power output can be derived using the equations of conservation of mass, momentum and energy (for an incompressible flow)[2]:

$$\dot{m} = \rho A_\infty u_\infty = \rho A_r u_d = \rho A_w u_w \tag{2.2}$$

$$T = \dot{m}(u_\infty - u_w) = (P_r^+ - P_r^-)A_r \tag{2.3}$$

$$E = \frac{1}{2}m(u_\infty^2 - u_w^2) \tag{2.4}$$

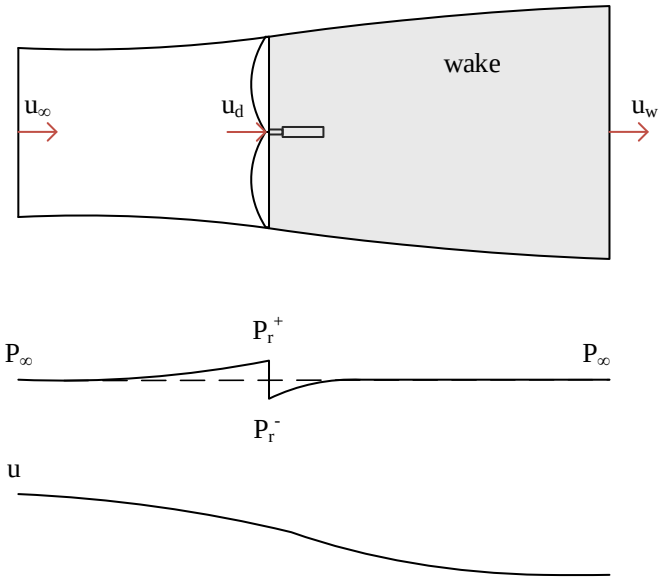

Figure 2.1 Actuator disk theory

The power extracted by the wind turbine is equal to the power performed by force T on the rotor; therefore, by equating (2.5) and (2.6), the expression for wind speed in front of the rotor can be found (2.7).

$$P_{turbine} = \frac{dE}{dt} = \frac{1}{2}\dot{m}(u_\infty^2 - u_w^2) \tag{2.5}$$

$$P_{turbine} = Tu_r = \dot{m}(u_\infty - u_w) \tag{2.6}$$

$$u_d = \frac{1}{2}(u_\infty + u_w) \tag{2.7}$$

The axial induction factor a, which is defined by the ratio between wind speed just in front of the rotor and freestream wind speed, is a term commonly used in literature.

$$a = 1 - \frac{u_d}{u_\infty} \tag{2.8}$$

In order to compare the performance of different turbines, the power coefficient C_p is defined. This coefficient is calculated as the ratio between the power extracted by the rotor, and the kinetic power of the wind stream. Said parameter can also be defined in terms of the axial induction factor (2.9). The optimal C_p value that maximizes the turbine power production, is found at $a = {}^1/_3$ which results in $C_{p,max} \approx 59\%$, which is known as the Betz limit.

$$C_p = \frac{P}{P_w} = \frac{P}{\frac{1}{2}\rho A_r u_\infty^3} = 4a(1-a)^2 \tag{2.9}$$

Similarly, the thrust coefficient that relates the thrust force and the wind kinematic force can be defined as shown in (2.10). When C_p is at its optimal value, C_T is equal to ${}^8/_9$.

$$C_T = \frac{T}{\frac{1}{2}\rho u_\infty^2 A_r} = 4a(1-a) \tag{2.10}$$

As a result of the energy conversion that takes place in the wind turbine rotor, a turbulent wind flow, known as wake, is generated downstream. This wind flow suffers a decrease in speed and an increase in turbulence intensity, which combined create the wake effect. Such phenomena negatively influences the performance of downstream turbines, resulting in reduced power production and increased fatigue loads. Both wind speed and turbulence intensity tend to their atmospheric values as the wind flow travels downstream.

However, the recovery of wind speed is faster than the decay in turbulence intensity. While velocity deficit is minimal after 10 rotor diameters (D) downstream, increased turbulence is still noticeable after 15 D. In wind farms, wind turbines are usually spaced between 6 D and 10 D, which means that wake effect is to be taken into account when evaluating and optimizing wind farm performance.

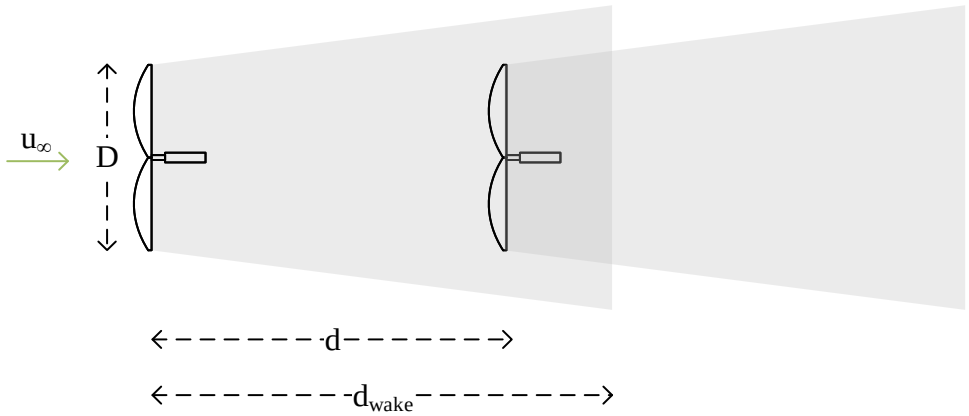

Figure 2.2 Wake effect

In order to describe wind speed variations inside the wake, several wake models have been proposed, such as Jensen [8] and Frandsen [9].

2.2. Wind Turbine Power Production

The power output of a wind turbine is mainly dependent on the average incoming wind speed. A standard power curve for a generic wind turbine is shown in Figure 2.3.

Under a certain wind speed threshold, known as cut-in speed u_c, the wind flow has insufficient kinetic energy to overcome the mechanical and electrical resistance of the rotor. As wind speed increases over the cut-in value, the power output gradually grows until reaching the rated wind speed u_r. When wind flow attains the rated wind speed, the generator produces the maximum electrical power it is capable of. This limit to the generator output is known as rated power output P_r. At higher wind speeds, in order to protect the generator, the wind turbine is set to maintain a constant power output. Finally, when wind speed reaches the cut-out value u_f, the turbine stops operating to prevent damages due to high forces on the rotor.

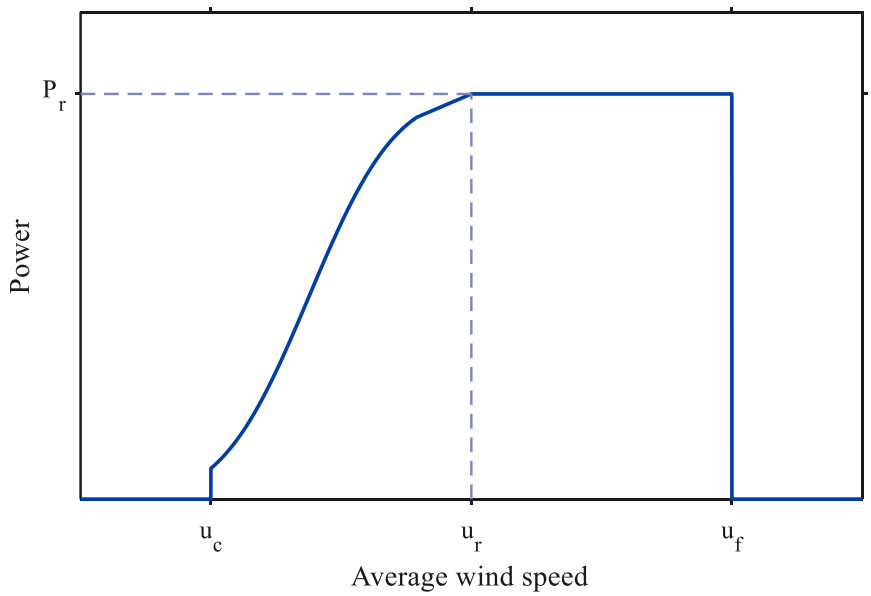

Figure 2.3 Wind turbine power curve

Grouping of wind turbines in wind farms leads to aerodynamic interactions that cause power losses and increased loads. This phenomena is a consequence of the wakes generated by turbines in the farm. Wake influence on turbine power production can be characterized by: turbine array geometry, ambient turbulence, terrain lay-out, wind-frequency distribution and operating settings of the turbines [2].

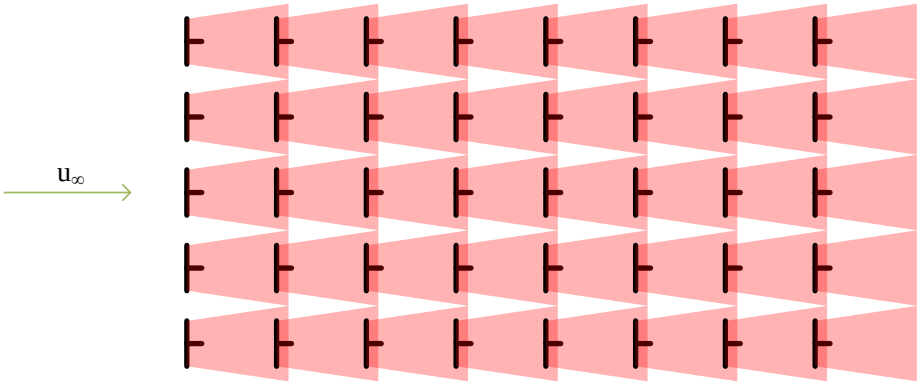

Figure 2.4 Wind farm wake effect

2.3. Wind Farm Control

The main objective of wind farm operation control is to meet network requirements with available wind energy. A general overview on farm control can be seen in Figure 2.5.

Wind farm control covers two main disciplines: power electronics engineering and mechanical/aerodynamic engineering. In this second field, the main points of focus are

maximizing total wind farm active power, following a power reference, and doing these tasks in a manner that reduces fatigue loads on the turbines[10].

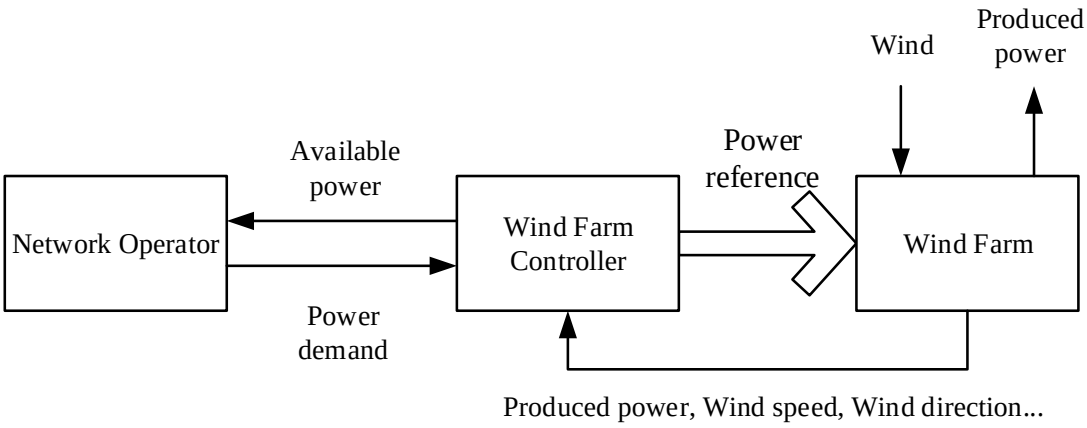

Figure 2.5 General Wind Farm Control [10]

Following a traditional control approach, each wind turbine operating point is set in a way that maximizes its own power production. This control strategy is known as greedy control strategy, since control actions do not account for the effects that the operation strategy has on downstream turbines. Because of aerodynamic interactions, this approach does not lead to maximizing power production for the whole farm. Two main control solutions exist when trying to optimize wind farm power output: power derating and yaw and tilt control [10]. In this thesis, a control solution that involves down-regulating turbine operation in order to maximize wind farm power production is investigated.

By derating a wind turbine, its power output is reduced, and the generated wake effect is weakened. In order to follow a derated power output reference, different control strategies can be used to down-regulate turbine operation, such as maximizing the rotational speed, maintaining a constant tip speed ratio or maintaining constant rotational speed. Further information on these techniques can be found in [11].

On the other hand, controlling yaw and tilt angles of the rotor has the effect of deflecting the wake from downstream turbines. By changing direction of the wake, wind speed deficit seen by downstream turbines becomes lower. Therefore, their power production is increased, which can potentially result in higher total wind farm power.

Using the previously mentioned wake models, a wind farm power function can be derived. Model-based approaches use this power function to determine the optimal control actions that maximize wind farm power output. However, this optimized solution

might not be the true optimum for the real wind farm due to the approximation present in the modelling procedure. Furthermore, deriving analytical power functions becomes more complex as the number of turbines in the target wind farm increases.

In order to overcome these difficulties, several data-driven approaches have been investigated. These techniques aim to maximize wind farm power production using only the control inputs and the power measurements, as shown in Figure 2.6, without needing an analytic expression of the wind farm power function. The following chapter presents a review of existing in-operation learning techniques that use a data-driven approach.

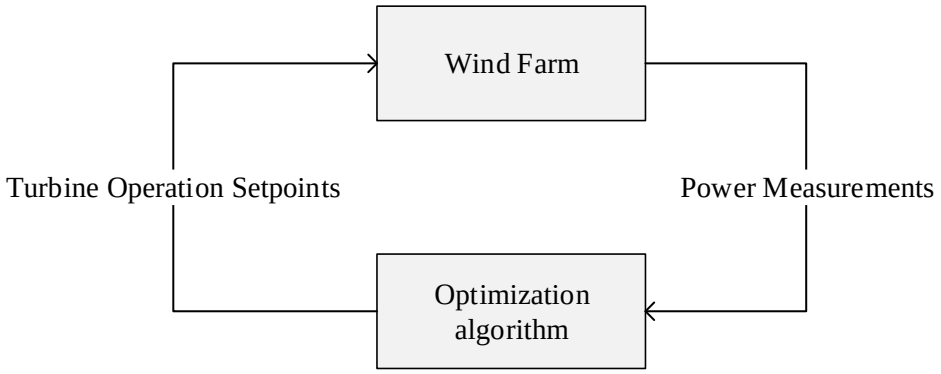

Figure 2.6 Data-driven optimization scheme

Chapter 3

Review of Existing In-Operation Learning Techniques

This chapter presents a literature review of different optimization algorithms that can be applied to the wind farm power maximization problem. Reviewed techniques involve finding the optimal operational strategy that coordinates wind turbines control actions, in order to optimize power production. Following this literature review, BA algorithm is chosen as the optimization algorithm to be implemented in the developed in-operation learning technique.

Under a conventional (non-cooperative) control strategy, each wind turbine aims to maximize its own power production, shown in equation (3.1). Consequently, every turbine sets its control actions (x) to 1/3 for the axial induction factor (a_i) and to 0º for the yaw angle (o) [2], regardless of the effect that this decision has on the power production of downstream turbines.

$$P_{farm} = \sum_{i=1}^{N} \max_{x_i} P_i(x) \tag{3.1}$$

On the other hand, a cooperative control approach seeks to maximize the total power production of the whole wind farm, as shown in equation (3.2). To accomplish this objective, the control algorithm has to account for the aerodynamic interactions between

turbines, such as wake effects. One option to deal with these interactions is to develop wake models and implement them in the control strategy. However, the complexity of these models makes them not suitable for real-time control of a wind farm. An alternative is to use a model-free control algorithm capable of adjusting control actions of the turbines in real-time, responding to changing wind conditions.

$$P_{farm} = \max_{x} \sum_{i=1}^{N} P_i(x) \tag{3.2}$$

In order to develop the latter strategy several methods have been proposed by different authors. These techniques can be categorized in two main groups, one containing those algorithms that employ stochastic procedures and the other comprising the ones using directed search strategies. A classification of the reviewed methods in said categories is shown in Table 3.1.

Table 3.1 Reviewed optimization techniques classification

Stochastic search	Directed search

• Game Theoretic methods:
  ○ Safe Experimentation Dynamics
  ○ Pay-Off Based Distributed Learning for Pareto Optimality
• Decentralized Discrete Adaptive Filtering methods:
  ○ Decentralized Aggressive Discrete Stochastic Approximation
	• Maximum Power-Point Tracking methods:
	  ○ Gradient-Ascent MPPT method
	  ○ Quasi-Newton MPPT method
	• Bayesian Optimization methods:
	  ○ Bayesian Ascent algorithm
	• Decentralized Discrete Adaptive Filtering methods:
	  ○ Decentralized Regret-Based Adaptive Filtering
	• Particle Swarm Optimization

3.1. Game Theoretic Methods

The application of game theoretic learning algorithms and cooperative control to the problem of total wind farm power production was proposed by Marden et al.[3]. Specifically, two possible implementations were described: Safe Experimentation Dynamics (SED) and Pay-Off Based Distributed Learning for Pareto Optimality (PDLPO). The main difference between both algorithms lies in the amount of information

available to the individual wind turbines. SED assumes full communication between all the turbines in the wind farm. On the other hand, PDLPO, limits the information exchange range.

3.1.1. Safe Experimentation Dynamics

The SED algorithm aims to maximize the sum of the power productions of the turbines in the wind farm (3.3). This approach requires full communication, meaning that every turbine has to know the total power production of the wind farm at any given iteration.

$$\max P(a) = \sum_{i \in N} P_i(a_i) \tag{3.3}$$

The performance of each turbine is controlled by a local state variable that is represented by $[a, P]$, where a_i is the benchmark action (the axial induction factor of the turbine) and P is the benchmark power (the total power produced by the whole farm). After the initialization, each subsequent iteration selects a new control variable. The new input can be either the baseline action $\bar{a}_i$, or a new random action within a uniform distribution of a. The probability of exploring a new control variable is controlled by the parameter ε.

$$a_i(t) = \begin{cases} \bar{a}_i(t) \; with \; probability \; (1-\varepsilon) \\ RAND \; with \; probability \; \varepsilon \end{cases} \tag{3.4}$$

Then, the state variables of the turbine are updated by observing the results of the changes made. If the new a_i has yielded more power, the baseline action and the baseline power are updated accordingly.

$$\bar{a}_i(t+1) = \begin{cases} a_i\,, & P(a(t)) > \bar{P}_i(t) \\ \bar{a}_i\,, & P(a(t)) \leq \bar{P}_i(t) \end{cases} \tag{3.5}$$

$$\bar{P}_i(t+1) = \max\{P\big(a(t)\big), \bar{P}_i(t)\} \tag{3.6}$$

The application of this algorithm leads to a nondecreasing power production, since the state variables are only updated when the changes result in a better performance. However, given the probabilistic nature of the algorithm many iterations are needed before converging to an optimum.

3.1.2. Payoff-Based Distributed Learning for Pareto Optimality

The PDLPO algorithm, just like SED, can be used to maximize the total power production of the wind farm (3.3). This distributed learning algorithm only requires each turbine to have information about the behaviour of neighbouring turbines.

The performance of each turbine is controlled by a local state variable that is represented by $[a, P, m]$, which adds a new parameter, m, to the SED implementation. This parameter represents the mood of the turbine and can take two values content or discontent. The update law of the control actions is controlled by this new parameter.

Again the implementation of this algorithm leads to nondecreasing power production in the wind farm, but the number of iterations needed to reach optimality is even greater than with SED.

3.1.3. Algorithm Simulation Tests

In [3], the described algorithms were tested in two simulation scenarios using a wind farm power function derived from the Jensen wake model. As specified in the formulation of the algorithm, the optimization strategy used in the simulations is based on controlling the axial induction factor.

In the first scenario, power optimization of a 3-turbine row was performed using both SED and PDLPO algorithms. The number of iterations needed for the SED method to reach 95 % of the analytical maximum was around 400. In the case of PDLPO, convergence was much slower, needing up to 150.000 iterations to reach the same value. The second scenario further explored the optimization capabilities of SED in an 80-turbine wind farm layout. In this simulation, around 4.000 iterations were needed to reach 95 % of the analytical maximum.

Consequently, the large amount of iterations needed to optimize the target system make these algorithms not suitable for real time maximization of wind farm power production.

3.2. Maximum Power-Point Tracking Method

The maximum power-point tracking (MPPT) method is based on observing the changes in power production as a result of changes in the control actions, and then modifying the control parameters in the direction that yields a power increase. Gebraad and van Wingerden [4] proposed two possible implementations of this technique for the wind farm

power maximization problem: Gradient-ascent MPTT method (GA-MPPT) and Quasi-Newton MPPT method (QN-MPPT).

The objective function for the gradient methods is the efficiency by which the kinetic energy of the wind is converted into electrical energy, as shown in equation (3.7). Setting the conversion efficiency as the target of the optimization causes the control method to be adaptive to varying wind speeds.

$$max \sum_{i=1}^{n} \tilde{P}_i(a_1, a_2, \dots, a_i) \ , with \ \tilde{P}_i = \frac{P_i}{u_\infty^3} \tag{3.7}$$

The main barrier to the implementation of these methods for real time control strategies is the amount of time needed to observe the variation in wind farm power production due to the changes in control actions, which is mandatory in order to determine the gradient. The origin of this delay is the time needed for the wind flow to travel from one upstream turbine to downstream turbines. During this time, the variation in wind conditions will affect the evaluation of the efficiency.

Two measures are taken in order to overcome this challenge. First of all, the algorithm only accounts for the wake effect on the closest neighbouring turbine which is a reasonable approximation since the influence of the change in a control action is far larger in the closest turbine. Secondly, a delay is applied to the wind speed is taken into account when computing the efficiencies of the turbines. Given a certain freestream wind speed u_∞ measured in the upstream point x_{u_∞}, the delay $T_{u_\infty \to i}$ is defined as the time needed for the wind flow to travel from the measurement point (x_{u_∞}) to the location of a certain turbine (x_i). The wind speed for a turbine at a given instant is calculated as shown in equation (3.8). Therefore, the conversion efficiency is defined by the delayed wind speed (3.9).

$$u_{\infty,i}^{del}(\mathrm{t}) = u_\infty(t - T_{U_\infty \to i}) \tag{3.8}$$

$$\tilde{P}_i = \frac{P_i(t)}{u_{\infty,i}^{del}(t)^3} \tag{3.9}$$

3.2.1. Gradient-Ascent Method

The first implementation of the MPPT method is GA-MPPT, in which the control variable is continuously updated at every iteration k using the direction given by the gradient (3.10).

$$a_i(k+1) = a_i(k) + K\left(\frac{\partial \tilde{P}_i}{\partial a_i}(k) + \frac{\partial \tilde{P}_{i+1}}{\partial a_i}(k)\right) \tag{3.10}$$

The parameter K adjusts the size of the steps on the control variable and can be used to tune the convergence properties of the algorithm. Given that the conversion efficiency function $\tilde{P}_i$ is not known, gradients can be approximated from past iterations through first-order backward differencing (3.11).

$$\frac{\partial \tilde{P}_j}{\partial a_i}(k) \approx \frac{\tilde{P}_j(k) - \tilde{P}_j(k-1)}{a_i(k) - a_i(k-1)} \tag{3.11}$$

3.2.2. Quasi-Newton Method

The QN-MPPT algorithm improves the convergence properties of the previous implementation. In this case, the update law for the control parameter is as follows:

$$a(k+1) = a(k) + KB(k)J(k) \tag{3.12}$$

Where a is the vector containing the induction factors of all the turbines, $J(k)$ is an approximation of the Jacobian and $B(k)$ is an approximation of the inverse Hessian matrix. Again, in the calculation of the Jacobian matrix, the gradient is approximated using equation (3.13).

$$J(k) = \begin{bmatrix} \frac{\partial \tilde{P}_1}{\partial a_1}(k) + \frac{\partial \tilde{P}_2}{\partial a_1}(k) \\ \frac{\partial \tilde{P}_2}{\partial a_2}(k) + \frac{\partial \tilde{P}_3}{\partial a_2}(k) \\ \vdots \\ \frac{\partial \tilde{P}_{n-1}}{\partial a_{n-1}}(k) + \frac{\partial \tilde{P}_n}{\partial a_{n-1}}(k) \\ \frac{\partial \tilde{P}_n}{\partial a_n}(k) \end{bmatrix} \tag{3.13}$$

The matrix $B(k)$ which represents the inverse Hessian matrix and is approximated using the Davidon-Fletcher-Powell formula.

$$B(k) = B(k-1) + \frac{\Delta a(k)\Delta a(k)^T}{\Delta J(k)^T \Delta a(k)} - \frac{B(k-1)\Delta J(k)\Delta J(k)^T B(k-1)^T}{\Delta J(k)^T B(k-1)\Delta J(k)} \quad (3.14)$$

$$\Delta a(k) = a(k) - a(k-1) \quad (3.15)$$

$$\Delta J(k) = J(k) - J(k-1) \quad (3.16)$$

3.2.3. Algorithm Simulation Tests

In [4] the performance of MPPT algorithms, when optimizing the power production of a 60-turbine array, is compared against SED. The control variable used to perform the optimization was the axial induction factor. Conducted tests showed that both GA-MPPT and QN-MPPT perform better than SED for real-time control implementations.

According to the simulations, the power production increases that can be obtained with the MPPT approaches may be somewhat smaller than the obtained with the GT approach (although the differences are small). This a consequence of the gradient methods converging to a local maximum instead of an absolute maximum.

However, both MPPT control methods yield better time-efficiency, which is important for real time implementation. For instance, in the simulations the GT method needed up to 150 hours to reach the optimum, whereas the MPPT methods only needed 0,6 hours. The fact that the MPPT methods optimize the power conversion efficiency makes them suitable for time-varying wind speeds.

Comparing both MPPT methods, it is shown that the QN-MPPT method yields a higher power increase for the main wind direction (for which the convergence parameters are tuned). On the other hand, if the same parameters are maintained, the GA-MPPT is more robust and can adapt better to varying wind directions.

3.3. Bayesian Ascent Method

The BA method and its application to wind farm power maximization was proposed by Park and Law [5], [12], [13]. This algorithm is based on the Bayesian Optimization (BO)

technique with improvements to make it more adequate for its application on wind farm control.

BO techniques aim to control a physical system by learning about the target system (exploration), and then using this information to find the optimal input (exploitation). The exploration phase is carried out by iteratively approximating the relationship between inputs (x) and outputs (y) to a Gaussian process (GP) regression using all recorded data $D^n = \{(x^i, y^i)|i = 1, \dots, n\}$.

The estimated target function f is then used to find the inputs that improve the objective function values in the exploitation phase. Selection of the new input is performed by maximizing an acquisition function that incorporates both aspects of exploration and exploitation. One possible acquisition function is the expected improvement (EI), which chooses the next input based on the improvement obtained over the maximum historical value and is defined as:

$$x^{n+1} = \underset{x}{\operatorname{argmax}}\, EI(x) \triangleq E[max\{0, f(x) - f^{max}\}|D^n] \qquad x^l \leq x \leq x^u \tag{3.17}$$

Where f^{max} is the maximum target function value is in the historical data, $max\{0, f(x) - f^{max}\}$ is the improvement towards the optimum and $[x^l, x^u]$ are the boundaries of the control action.

The application of BO to the wind farm optimization problem has to overcome two main issues: the difference between two successive inputs and the uncertainty in the target function. First of all, it is not desirable to abruptly change the control actions of a physical system, due to limitations in its performance and stability. Secondly, the selection of new inputs from regions with high uncertainty can result in unexpected drops in the target function value.

The BA algorithm solves this issues by incorporating a trust region constraint from which new inputs have to be sampled. This trust region is defined as a hypercube centred on the input that has produced the best power output so far. If the new selected input improves the objective function beyond a certain threshold, the trust region grows in order to increase the learning capabilities of the algorithm. This dynamic adjustment of the trust region size helps to improve the convergence properties of the algorithm and ensures that

the inputs change gradually. Incorporating the trust region T constraint, the acquisition optimization problem is defined as follows:

$$x^{n+1} = \underset{x}{\operatorname{argmax}}\, EI(x) \triangleq E[max\{0, f(x) - f^{max}\}|D^n]$$
$$x^l \leq x \leq x^u \tag{3.18}$$
$$x \in T \triangleq \{x|\ \|x_i - x_i^{\max}\| < \tau_i\ for\ i = 1, \dots, m\}$$

Where x_i^{max} is the control variable value that yields f^{max} and τ_i is the trust region size for each control variable.

3.3.1. Algorithm Simulation Tests

The applicability of the BA method to the wind farm control problem was investigated in [12]. Simulations of a 25 turbines and 100 turbines wind farm suggest that the BA algorithm can effectively locate a local optimum with a small amount of samplings reaching values close to the analytical maximum in less than 50 iterations, which is useful for its application in real time control strategies. It was also shown that this algorithm can monotonically improve the power production of the wind farm. In these simulations, a continuous wake model based on the Jensen model was used to obtain the analytical wind farm power function, which was used as the target function for optimization. As well as in simulation tests, BA algorithm also proved to be able to maximize power production in scaled wind tunnel experiments. The control approach used in these tests combined induction-based and yaw-misalignment strategies.

Furthermore, BA method was tested under different sets of varying parameters, such as the number of wind turbines in the wake, the wind direction and the distance between turbines. Results showed that BA manages to improve the energy conversion efficiency over the greedy control strategy under all circumstances. However, the magnitude of this improvement greatly changed in certain situations. For instance, a change in the wind direction that directed the wake away from the downstream turbine resulted in a smaller gain over the greedy strategy, since the wake effect became less relevant.

3.4. Decentralized Discrete Adaptive Filtering Methods

The decentralized discrete adaptive filtering methods use a distributed control approach in which each turbine has limited range to exchange information with neighbouring

turbines. This approach to wind farm control eliminates the need of a centralized control unit that collects information from all the turbines and sends the adequate commands. However, this also means that the maximized function is not the wind farm power production, but the local power production, defined as:

$$Q_i(a_i) = P_i(a_i)|a_{-i} + \sum_{j \in N_i} P_j|a_{-j} \tag{3.19}$$

Where Q_i is the local power measured by turbine i, calculated by adding its own power production P_i, plus the power production of the set N_i of neighboring turbines with which the current turbine can communicate, a_i is the control action for turbine and a_{-i} is the set of control actions of upstream turbines. Therefore, the goal of the decentralized control strategy is to find the set of control actions a^* that maximizes the local power production of all the turbines.

Two algorithms implementing this strategy have been proposed by Zhong and Wang [14]: decentralized aggressive discrete stochastic approximation (DADSA) and decentralized regret-based adaptive filtering (DRAF). The first algorithm employs a stochastic search to determine the optimum control actions, whereas the second proposed algorithm incorporates a directed search scheme based on a regret value. Both algorithms are run on each turbine independently.

3.4.1. Decentralized Aggressive Discrete Stochastic Approximation

Given a discrete set $A = \{a_1, \dots, a_k\}$ of possible induction factors for each turbine and an initial induction factor a^0, the algorithm will iteratively sample new values from the set and evaluate its performance. In each iteration, the local power production of the baseline strategy is compared to the one obtained by setting a control action randomly sampled from A. If the new control action yields an improvement over the baseline strategy, this baseline strategy is updated.

A state occupation vector of the same length as A, stores a magnitude proportional to the amount of times each induction factor has been visited. As a result of keeping always the strategy that produces the most power as the baseline action, the induction factor that maximizes the local power production eventually becomes the most visited.

3.4.2. Decentralized Regret-Based Adaptive Filtering

The DRAF algorithm uses a similar principle to the DADSA algorithm. However, instead of selecting the values from the discrete set A uniformly, the selection is based on a regret value. This regret value is calculated taking into account the power production each value in A has yielded in previous iterations. In each iteration, the turbine is more likely to sample the candidate control action that have high regret values.

With this modification, the search for the optimum becomes directed and the control actions closer to the optimum are visited more frequently, thus improving the convergence rate of the algorithm.

3.4.3. Algorithm Simulation Tests

In [14] the performance of DADSA and DRAF is compared against the game theoretic method SED. Two scenarios are investigated, a 10 turbine-array and a 25-turbine array. In both cases, the target function for the optimization is obtained using the Jensen wake model, and the axial induction factor is employed as control variable.

It is shown that DADSA and DRAF can find an optimum value (close to the real optimum) under time varying conditions, which is not possible using the SED algorithm. Simulations prove that the convergence rate for DADSA and DRAF is faster than SED while managing to obtain a higher power production value.

3.5. Particle Swarm Optimization

The last optimization technique reviewed is Particle Swarm Optimization (PSO), an algorithm originally intended for simulating social behaviour. The main advantage of PSO is its simple implementation and low computational cost. This algorithm has no yet been proposed as a possible optimization technique for the wind farm power production problem. However, a possible implementation is studied based on the reviews of this algorithm published by [15], [16].

Even though the resulting algorithm involves only simple calculations that would guarantee fast iterations, the performance of the technique would be limited by the physical system response time. This is a consequence of the time needed to observe the effect of a control action on the power production.

3.6. Viability Evaluation of Reviewed Techniques

In this chapter an overview on different in-operation learning techniques has been presented. All the reviewed techniques are model-free and can be applied to the wind farm power maximization problem. However, the viability of the implementation to a real-time control structure, which is the objective of this thesis, differs between them.

First of all, the game theoretic methods (SED and PDLPO) both have been proved to require a large number of iterations before reaching optimality, which is caused by the randomness introduced in the selection of the control variable in each iteration. Even though this trial and error approach manages to converge to the global maximum for a sufficiently large number of iterations, the time limitations of wind conditions changes make it not suitable for real-time control.

As for the MPPT techniques reviewed, both GA-MPPT and QN-MPPT show a 300 times faster convergence rate over the game theoretic methods. These techniques have been proven to require as little as 0,6 hours to reach values close to the optimal. Such improvement is due to the introduction of the gradient in the update law of the control actions, replacing the use of stochastic search in game theoretic methods. The approximation of the gradient of the target function is performed using only the last iterations, thus making the algorithm require low computational power. Given that these techniques are local optimization techniques, the solution might converge to a local maximum instead of a global maximum. However, using the result of a model based optimization as starting condition can ensure that the solution obtained yields an improvement over the classical control strategy.

Regarding the BA optimization technique, results show a remarkably fast convergence to the optimum (under 20 iterations to reach 90% of the optimum). This characteristic is a result of the fact that BA stores all the information from previous iterations, thus giving a better approximation of the target function. As the amount of information grows, the algorithm procedure becomes inherently slower, but this becomes less relevant if the optimization is performed in a small number of iterations. The usage of a GP to estimate the wind farm efficiency function, means that the optimization procedure can work even with uncertainty present in the measured data. The existence of a trust region constraint in the sampling of new inputs improves the convergence capabilities of the algorithm and

ensures an almost monotonic increase of the target function. On the other hand, this constraint turns the BO procedure into a local optimization technique.

Finally, three decentralized algorithms have been considered DADSA, DRAF and PSO. These approaches manage to optimize the power production of the wind farm with limited communication between the turbines. However, in the case of DADSA and PSO, their stochastic nature hinders the convergence speed. For both decentralized discrete adaptive filtering methods, DADSA and DRAF, the usage of a discrete set of control actions means that the performance of the algorithm is dependent on an initial calibration. If there is no prior estimation of the most adequate control actions, it is a challenge to determine the sampling set, especially as the number of wind turbines grows.

In conclusion, the review of in-operation learning techniques has shown that the most adequate approaches for wind farm power production optimization are the MPPT methods and the BA methods. These techniques have been proven to provide a fast convergence and robust performance under time varying wind conditions.

3.7. Selection of Learning Technique

The objective of this literature review was to determine the most adequate learning algorithm to be implemented in a wind farm controller. The criteria for the selection is based on the performance of the algorithm and its applicability to real-time control.

The BA algorithm has been shown to be the best performing algorithm of the presented set. The optimization procedure of this algorithm is based on a directed search strategy that makes use of historical performance data. Such approach allows BA to find the optimal operating points requiring few iteration steps. Furthermore, as a result of the trust region constraint, this optimization is performed with gradual changes in the control actions and an almost monotonic increase of the target function.

The stochastic nature of the wind flow results in uncertain power output readings from the wind turbines. Consequently, another key feature of the BA optimization technique when applied to the wind farm power maximization problem, is its capability to optimize the target function with uncertainty present in the measured data.

Improvements to the application of this method, in order to obtain better performance in real time operation control of wind farms, are presented in this thesis.

Chapter 4

Bayesian Ascent Algorithm Implementation

This chapter presents the proposed implementation of the BA algorithm used in the in-operation learning technique. The developed technique must be able to meet the requirements presented in the objectives of the thesis, which include applicability to real-time wind farm control and robustness under time-varying conditions

The proposed approach is based on the BA algorithm developed by Park and Law [12], [13] that is discussed in the previous chapter.

4.1. Wind Farm Control Problem Definition

The proposed implementation of the BA algorithm aims to maximize the total power production of the wind farm by individually controlling the operation conditions of the turbines. This control strategy must account for the interactions between the turbines in order to find the optimal operation conditions. Furthermore, it is mandatory for the algorithm to perform this optimization in a time efficient manner that ensures its applicability to real-time control.

In the implemented optimization procedure, optimal operating conditions are defined as those that maximize the efficiency by which the wind farm converts incoming wind power into electrical power. This normalization used in the definition of the target function results in an optimization solution that is independent from wind speed (in a given range). Consequently, the objective function for the optimization procedure is defined as:

$$\max f(x) = \frac{1}{N}\sum_{i=1}^{N} \eta_i\left(C_p, \theta^w\right) \tag{4.1}$$

Where N is the number of wind turbines, C_p is the power coefficient, θ^w is the wind direction and η_i is the efficiency of the conversion in each wind turbine. This efficiency is calculated using the freestream wind speed u_∞, as shown in (4.2) (4.3).

$$\eta_i = \frac{P_i\left(C_p, u, \theta^w\right)}{P_{wind}} = \frac{P_i\left(C_p, u, \theta^w\right)}{\frac{1}{2}\rho A_r u_\infty^3} \tag{4.2}$$

$$P_i = \frac{1}{2}\rho A u^3 C_p \tag{4.3}$$

Possible optimization decision variables are the turbine induction factor and the turbine yaw angle. Yaw angle-based optimized operation strategies typically yield a larger increase in total wind farm power. However, such strategies are challenging due to the impact of yaw-misalignment on wind turbine mechanical loads. Induction-based operation strategies, on the other hand, can be implemented within the current certified operational envelope of wind turbines. Therefore, this latter strategies can be more easily implemented in present wind farms and hence in the focus of this work. Consequently, it is assumed that all the turbines are oriented perpendicular to the wind direction and, consequently, the yaw angle o is taken as a constant and equal to 0.

Following the chosen approach, upstream turbines are downregulated in order to allow for larger power production in downstream turbines. The implementation of this strategy only requires a single control variable x_i to be set for each turbine, resulting in a simpler optimization objective function.

$$\max f(x) = \frac{1}{N}\sum_{i=1}^{N} \eta_i(x_i) \tag{4.4}$$

In this thesis, the control variable used to set the operating points of the different turbines in the wind farm is defined as the derating factor. This parameter quantifies how much the power production of the turbine is reduced below the power output that could be obtained, if the turbine was maximizing its own power production i.e., when applying a greedy control approach.

Therefore, the derating factor is formulated as the ratio between the current turbine power output and the maximum power output with the current incident wind speed (Figure 4.1).

$$DerFact_i = \frac{P_{d,i}}{P_{t,i}} \tag{4.5}$$

Consequently, the aim of the optimization is to find the set of derating factors, x, that maximizes total wind farm power production. When performing the maximization, the derating factor of the last turbine in the array is always set to 1, since the greedy control action is the global optimal strategy when there is no need to account for turbines in the wake.

$$x = (DerFact_1, DerFact_2, \dots, DerFact_n) \tag{4.6}$$

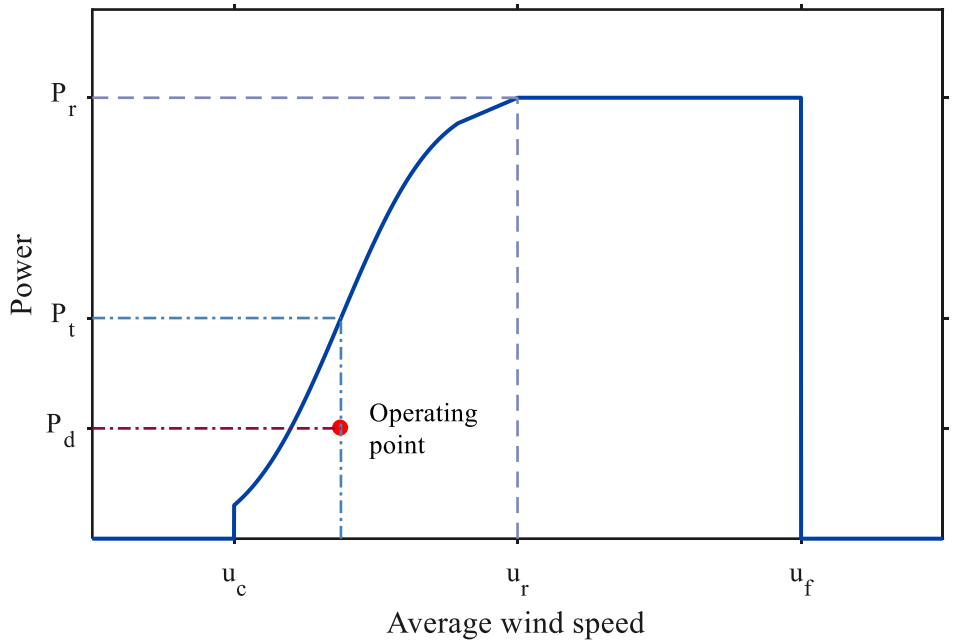

Figure 4.1 Wind turbine down-regulation

4.2. Bayesian Ascent Algorithm

The BA algorithm is an adaptation of BO to the in-operational maximization of wind farm total power. BO is a model-free, data driven optimization approach that can be applied to find the optimal operational conditions of the described wind farm power maximization problem. Said optimization technique relies on constructing a GP regression that fits the control inputs to the measured noisy outputs, which then can be used to derive the optimal values for the turbine control actions.

BA algorithm further improves the applicability of the BO technique to real-time wind farm power production maximization by adding a trust region constraint in the sampling of new inputs. Such modification ensures that the algorithm is able to improve the target function rapidly and with few measurements of data.

4.2.1. Bayesian Optimization

BO seeks to find the input $x = (x_1, x_2, \dots, x_m)$ that solves $x = \text{argmax}\, f(x)$. The main difference with other optimization procedures is that BO constructs a probabilistic model for the objective function $f(x)$ and then exploits this model to select the new inputs that optimize it. In contrast with gradient and Hessian optimization algorithms, BO makes use of all the information from previous evaluations of the objective function. As a result, BO is able to find the optimal input using a low number of evaluations, making this technique especially interesting when the objective function is expensive to evaluate. On the other hand, the procedure for the selection of new inputs involves extended computational time compared to gradient techniques.

When a new input X is selected, the corresponding noisy response y is observed. The analytical expression of the objective function is unknown, but it is modelled using a GP regression. The variable $\in$ represents the uncertainty, which is assumed to follow a Gaussian distribution.

$$y = f(x) + \in \tag{4.7}$$

As previously mentioned, BO aims to construct the regression model for the objective function, while maximizing it at the same time. In order to do so, each iteration of the algorithm consists of two phases, learning and optimization.

In the learning phase, the objective function $f(x)$ is modelled with a GP regression $p(f^{1:n}) = GP\big(\mu(\cdot), k(\cdot,\cdot)\big)$, constructed using all previous inputs $x^{1:n} = \{x^1, \dots, x^n\}$ and observed outputs $y^{1:n} = \{y^1, \dots, y^n\}$. The result obtained from the regression is a mean function $\mu(x)$, which describes the overall trend in the function values and a kernel function $k(x, x')$, which is used to approximate the covariance in function values. If there is no previous knowledge of the target function, $\mu(x)$ is set to 0 in order to simplify the learning procedure. The likelihood function of the measured output is represented as a Gaussian distribution, with a mean equal to the value obtained

in the regression followed by a noise variance σ_e^2 that quantifies the level of noise assumed to exist in the measurements.

$$p(y^{1:n}|f^{1:n}) = N(f^{1:n}, \sigma_e^2 I) \tag{4.8}$$

Given the regression model, the function value $f(x)$ for an unseen input x and previous observed outputs $y^{1:n}$ follows a multivariate Gaussian distribution [17]:

$$\begin{bmatrix} y^{1:n} \\ f \end{bmatrix} \sim N\left(0, \begin{bmatrix} K + \sigma_e^2 I & k \\ k^T & k(x,x) \end{bmatrix}\right) \tag{4.9}$$

Where $k^T = \left(k(x^1, x), \dots, k(x^n, x)\right)$ and K is the covariance matrix whose (i,j) entry quantifies the similarity between the inputs x^i and x^j. A high covariance value means that two inputs are strongly correlated, in contrast, the more two inputs differ the closer the covariance value is to zero. The (i,j) entry of this covariance matrix is evaluated using a squared exponential covariance function [17]:

$$K_{ij} = k(x^i, x^j) = \sigma_S^2 \exp\left(-\frac{1}{2}(x^i - x^j)^T diag(\lambda)(x^i - x^j)\right) \tag{4.10}$$

Which is described by the parameters σ_s and λ. The term σ_s is the signal variance that quantifies the overall magnitude of the covariance value. The vector $\lambda = (\lambda_1, \dots, \lambda_m)$ contains the characteristic length scales that compute the relevancy of each component of the input $x = (x_1, \dots, x_m)$ in predicting the response y.

Even though the GP is not a parametric model, the computation of the mean and covariance requires knowledge of σ_e, σ_S and λ, which are known as hyperparameters. In each iteration, these hyperparameters $\theta = (\sigma_e, \sigma_S, \lambda)$ are determined as the ones maximizing the log-likelihood of the historical data $D^n = \{(x^i, y^i | i = 1, \dots, n\}$.

$$\begin{aligned} \theta^* = \underset{\theta}{argmax}\, log(p(y^{1:n})) &= \underset{\theta}{argmax}\, log\left(\left(p(y^{1:n})\right)\middle| x^{1:n}, \theta\right) \\ &= \underset{\theta}{argmax}\, log\left(-\frac{1}{2}(y^{1:n})^T(K + \sigma_\epsilon^2 I)^{-1} y^{1:n}\right. \\ &\left. -\frac{1}{2} log|K + \sigma_e^2 I| - \frac{n}{2} log(2\pi)\right) \end{aligned} \tag{4.11}$$

Once the hyperparameters are optimized, the mean (4.12) and variance (4.13) functions that describe the GP regression can be calculated. These functions are then used to evaluate the value of the objective function corresponding to an unobserved input x.

$$\mu(x|D^n) = k^T(K + \sigma_e^2 I)^{-1} y^{1:n} \tag{4.12}$$

$$\sigma^2(x|D^n) = k(x,x) - k^T(K + \sigma_e^2 I)^{-1} k \tag{4.13}$$

In conclusion, given an input x, instead of returning a scalar value for the objective function, GP returns the mean and variance of a normal distribution over its possible values at x.

Once the learning phase is completed, the computed mean and variance functions are used to select the next input x^{n+1}, that combines learning more about the target function (exploration) as well maximizing its value (exploitation). In order to select the new sampling point, an acquisition function that incorporates both aspects of exploration and exploitation is used. Several acquisition functions have been proposed, but the expected improvement (EI) criterion has shown to be the best behaved [18]. The EI function is defined as:

$$x^{n+1} = \underset{x}{argmax}\, EI(x) \triangleq E[max\{0, f(x) - f^{max}\}|D^n] \tag{4.14}$$

Where $max\{0, f(x) - f^{max}\}$ is the improvement towards the maximum output compared to the maximum target function value f^{max}. The maximum target function value f^{max} is estimated in each iteration using the mean function instead of the maximum observed output y^{max}. This is due to the large amount of noise that might be present in the measurement value, which could interfere with the sampling procedure.

$$f^{max} = \max_{x \in x^{1:n}} \mu(x|D^n) \tag{4.15}$$

Under the GP model, the EI can be computed in closed form:

$$EI(x) = \begin{cases} (\mu(x) - f^{max})\Phi(Z) + \sigma(x)\emptyset(Z), & \sigma(x) > 0 \\ 0, & \sigma(x) = 0 \end{cases} \tag{4.16}$$

$$Z = \frac{\mu(x) - f^{max}}{\sigma(x)} \tag{4.17}$$

Where $\Phi(Z)$ and $\emptyset(Z)$ are the cumulative distribution and probability density functions of the standard normal distribution.

The $EI(x)$ function has a high value at a point x when, either the mean $\mu(x)$ or the variance $\sigma(x)$ are large. Therefore, by selecting an input that maximizes $EI(x)$, the

sampled input will come from a region where a high value of the objective function is expected, or from an input region that is yet to be explored.

4.2.2. Bayesian Ascent

Even though BO algorithm can optimize a target function using a limited number of sampled points and function evaluations, two issues arise when trying to control a physical system. First of all, the difference between two successive sampled inputs can be too large, producing undesired abrupt changes in the control actions. Secondly, inputs might also be sampled from a region where the uncertainty is too high, leading to sudden drops in target function values.

In order to handle these challenges, Park & Law proposed a modified sampling strategy for the BO algorithm, creating the BA algorithm [5],[12],[13]. Specifically, a trust region constraint is imposed when sampling new inputs using the EI acquisition function. Such constraint means that the next solution x^{n+1} is chosen near the best solution observed so far (x^{max}, f^{max}), expediting the rate of convergence to a (local) optimum and avoiding drastic changes to the control actions.

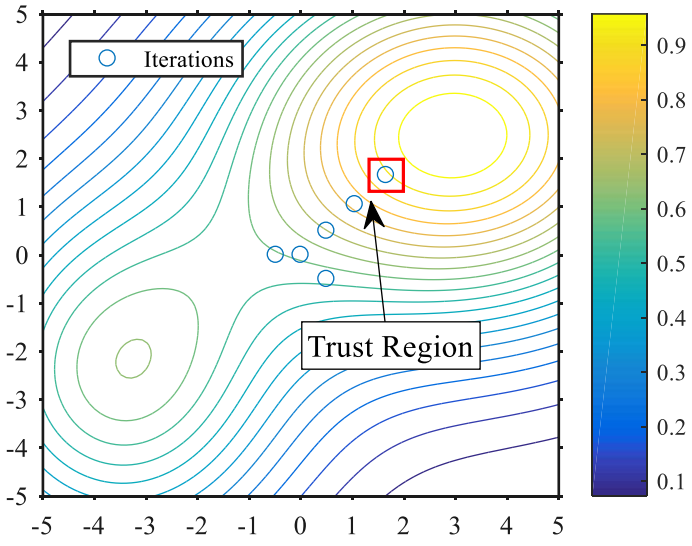

Figure 4.2 Trust region sampling

With the addition of a trust region, the sampling of new inputs using the EI acquisition function becomes a constrained optimization problem.

$$x^{n+1} = \underset{x \in A \cap T}{argmax}\, EI(x) \triangleq E[max\{0, f(x) - f^{max}\} | D^n] \tag{4.18}$$

$$x \in A \triangleq \{x |\, x^l \leq x \leq x^u\}$$

$$x \in T \triangleq \{x |\, \|x_i - x_i^{max}\| < \tau_i \; for \; i = 1, \dots, m\}$$

Where A is the region defined by the absolute bounds of the control actions and T is the trust region built around the best observed solution. This trust region is defined as a hypercube, its centre being x^{max} , and its size being controlled by τ. Each component τ_i determines the range where the x_i component of the solution is being sampled next. Size of the trust region is adjusted in each iteration to ensure a monotonic increase in the target function and gradual convergence. Such adjustment means that the trust region grows when the sampled input improves the previous maximum observed output over a certain threshold. On the other hand, if this condition is not met, the trust region size is reset to its initial value. The trust region update law at an iteration n, is summarized as:

$$if \; y^{n+1} - f^{max} \geq \frac{\gamma}{n}(f^{max} - y^1) \; then$$

$$\tau^{n+1} = \beta\tau^n$$

$$else \tag{4.19}$$

$$\tau^{n+1} = \tau^1$$

$$end \; if$$

$$\beta > 1 \tag{4.20}$$

Where y^{n+1} and f^{max} are the current observed output and maximum observed output, n is the number of iterations, γ is a parameter to control the average improvement threshold, and β is the trust region growth coefficient. For the optimization, γ is set to 0,05 and β is set to 1,1.

4.3. Strategy for Varying Wind Conditions

Given the nature of wind farm operation, robustness under varying wind conditions is a mandatory characteristic for any control strategy. The developed in-operation learning technique includes a strategy to handle variations in different wind parameters, namely wind speed, wind direction and turbulence intensity.

First of all, by normalizing the power output of the turbines using the incoming freestream wind speed, the resulting objective function is independent from wind speed changes

within a certain wind speed range. This normalization is true for speed below the rated wind speed, and valid a range of wind speeds where C_T is constant. Therefore, the same optimal control strategy can be applied within this wind speed range.

Secondly, binning is used to handle changes in wind direction and turbulence intensity and wind speeds outside above discussion wind speed range. This strategy involves dividing the range of values that a wind parameter can take in a certain number of intervals, therefore reducing the variability that a change in the wind condition introduces in the target function. Size of the binning intervals has to be adjusted in order to ensure that the optimization solution is valid in all the interval ranges. The proposed binning strategy can be seen in Table 4.1.

Table 4.1 Wind condition binning strategy

Wind condition	Range	Interval width	Number of intervals
Wind direction	0-360º	5º	72
Turbulence intensity	4 %- 10 %	1 %	6

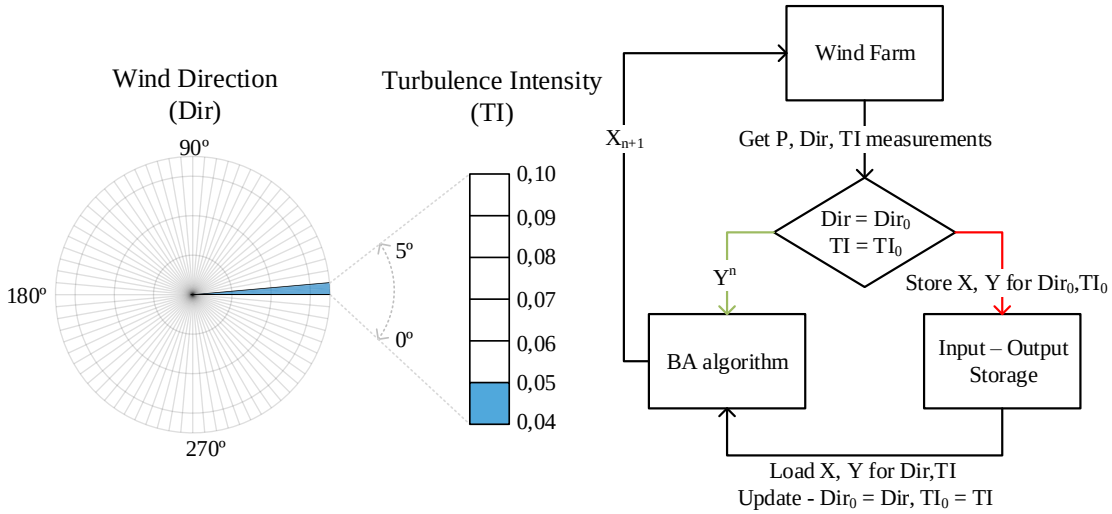

Figure 4.3 Changing wind conditions strategy

Following this binning strategy, the suggested approach to deal with variations in these two wind conditions is to run multiple BA algorithms in parallel that can be started and stopped when a change in wind conditions is observed. Each algorithm optimizes the operation settings of the wind farm for one of the bins, in which the variables have been divided. Since both the wind direction and the turbulence intensity can vary

independently, the total number of optimization algorithms to run is 432 in order to cover all the possible combinations.

Given the data-driven approach that the algorithm uses, storing the input X and output Y arrays is enough to resume the optimization at the point where it was stopped. When a change in wind direction or turbulence intensity is observed, the data for the current optimization is stored, while the data set for the new wind condition is loaded. Furthermore, since changes in wind conditions are continuous and gradual over time, it is safe to assume that the optimal strategy for a new wind condition is similar to the optimal solution for the previous wind condition. Therefore, it is reasonable to use the strategy optimized by the current BA algorithm as a starting point for the new wind conditions. An overview of the proposed strategy for dealing with wind condition changes is shown in Figure 4.3.

Chapter 5

Test of Algorithm on Analytical Functions

In this section, results of optimization tests with selected analytical functions are presented. The aim of these tests was to evaluate the performance of the BA algorithm, and to analyse the effect of its parameters in the optimization. In order to conduct this assessment, functions with different degrees of complexity were used, including quadratic, exponential and trigonometric functions, among other common optimization test problems. The functions used to test the algorithm were obtained from [12] and [19].

The obtained results suggest that the BA algorithm is able to successively optimize functions with up to 10 variables. Additionally, such optimization is performed with a limited number of function evaluations (under 20 iterations).

5.1. Quadratic Function

First of all, the optimization technique was tested using a simple two-variable quadratic function with a single global optimum (5.1).

$$f(X) = -\frac{(x_1^2 + x_2^2)}{50} + 1 \tag{5.1}$$

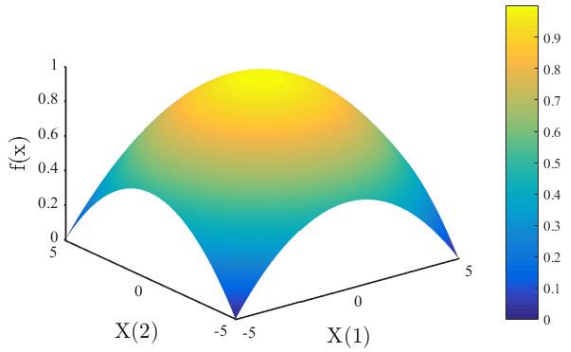

Figure 5.1 Quadratic function surface plot

BA algorithm manages to reach values above 95% of the analytical optimum in 7 iterations, as can be seen in Figure 5.2. Furthermore, the function is improved almost monotonically, which is a desirable characteristic for physical systems optimization. As Figure 5.3 shows, the algorithm tends to the optimum following the maximum improvement direction.

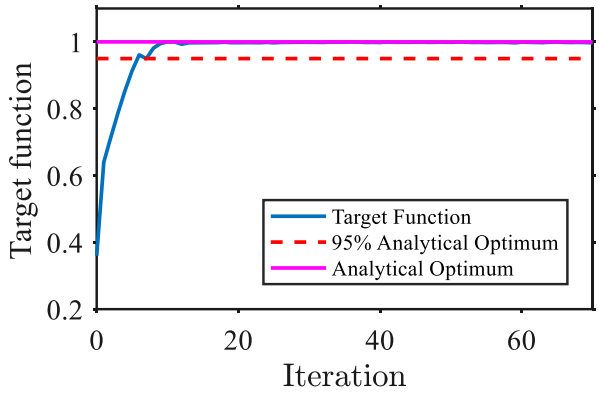

Figure 5.2 Quadratic function optimization

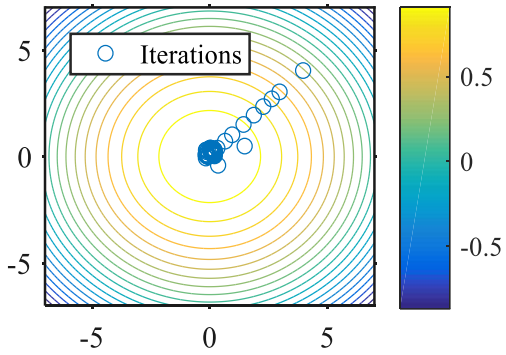

Figure 5.3 Quadratic function contour

5.2. Exponential Function

Once the technique was proven suitable for a simple single global optimum problem, the behaviour of the algorithm in a multiple local optima scenario was assessed. In order to perform this second test, an exponential function with two local optimums was used (5.2).

$$f(X) = \exp\left(-\frac{(x_1-3)^2}{22} - \frac{(x_2-2{,}5)^2}{16}\right) + 0{,}6\exp\left(-\frac{(x_1+3{,}5)^2}{12} - \frac{(x_2+2{,}5)^2}{16}\right) \tag{5.2}$$

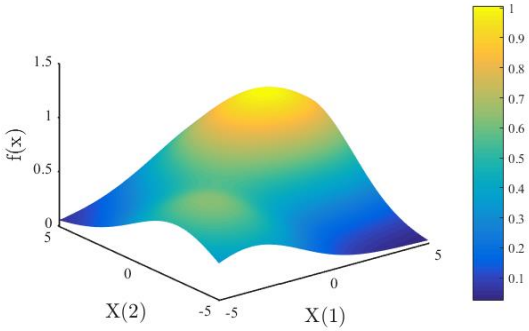

Figure 5.4 Exponential function surface plot

As in the previous case, the optimization technique manages to converge to a solution in less than 10 function evaluations. However, imposition of a trust region constraint and starting point choice, can potentially result in convergence to the lower local optimum (Figure 5.5 and Figure 5.6). Selecting an initial guess close to the higher optimum, allows the algorithm to converge to the higher optimum (Figure 5.7 and Figure 5.8).

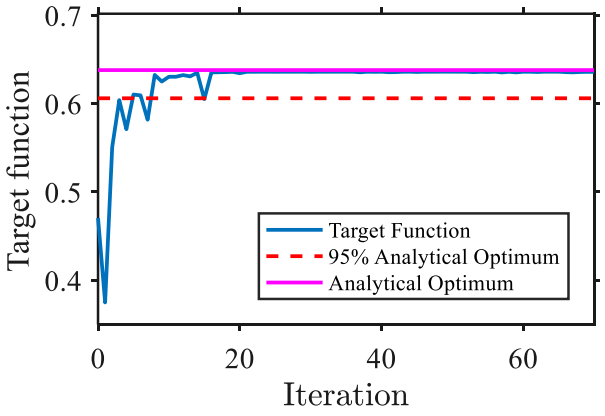

Figure 5.5 Exponential function optimization

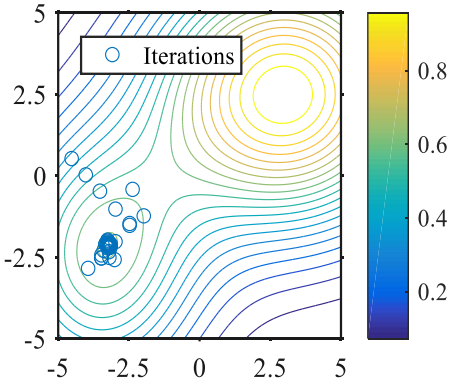

Figure 5.6 Exponential function contour

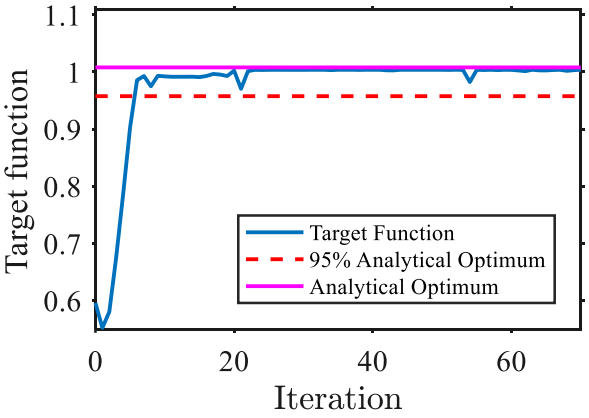

Figure 5.7 Exponential function optimization

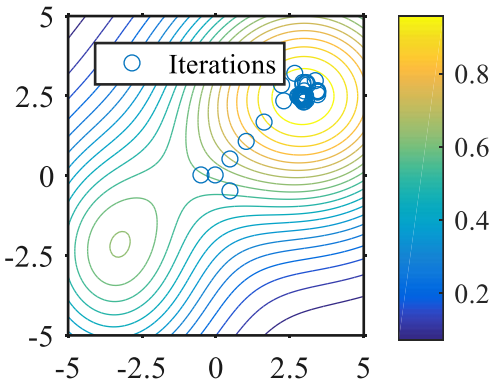

Figure 5.8 Exponential function contour

5.3. Trigonometric Function

To further investigate the influence of the starting point selection and trust region sizing, the BA technique was tested using a trigonometric function with a large number of local optimum (5.3).

$$f(X) = x_1 x_2 \sin(x_1) \sin(x_2) \tag{5.3}$$

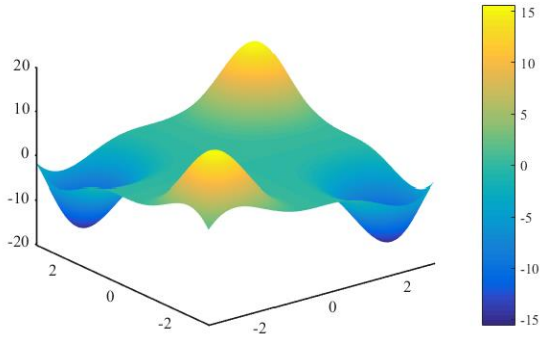

Figure 5.9 Trigonometric function surface plot

Adequate trust region sizing allows the algorithm to converge to a local optimum, as can be seen in Figure 5.10. On the other hand, an excessively large trust region negatively effects the convergence properties of the algorithm. In Figure 5.11, a test run of 70 iterations is shown. A larger trust region size results in the algorithm not being able to converge to a solution, this limitation is due to the high variability of the function evaluations in the search space. Consequently, it is proven that adequate trust region sizing is key to guarantee the convergence properties of the algorithm.

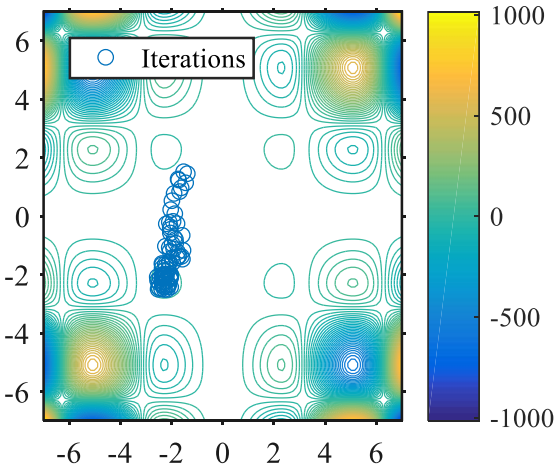

Figure 5.10 Trigonometric function contour

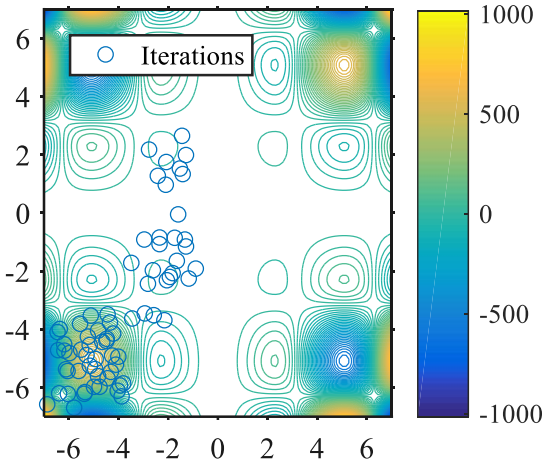

Figure 5.11 Trigonometric function contour

5.4. Wood Function

Following the conducted tests on two-variable functions, the relationship between the algorithm performance and an increased number of variables was investigated. Therefore, the BA was applied to optimization problems of increased complexity.

First of all, the optimization technique was applied to a four-variable function known as Wood function (5.4). Results show that, in spite of the greater number of variables, the BA algorithm is still able to reach 95% of the analytical maximum in 12 iterations (Figure 5.12).

$$\begin{aligned} f(X) = {} & 100(x_1^2 - x_2)^2 + (x_1 - 1)^2 + (x_3 - 1)^2 \\ & + 90(x_3^2 - x_4)^2 + 10{,}1((x_2 - 1)^2 + (x_4 - 1)^2) \\ & + 10{,}8(x_2 - 1)(x_4 - 1) \end{aligned} \tag{5.4}$$

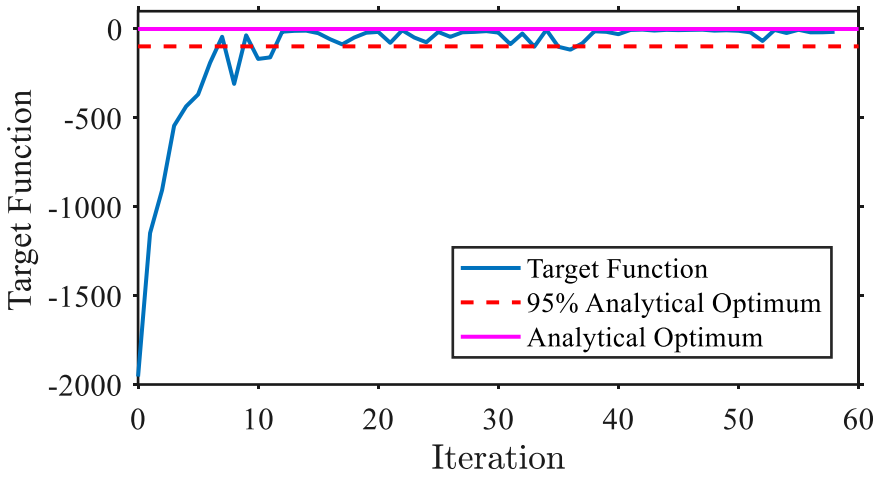

Figure 5.12 Wood function optimization

5.5. Broyden Function

Performance of the algorithm with a greater number of variables was further tested. In this case, an n-dimensional function in its seven variable version with $p = 7/3$, known as Broyden function (5.5), was used. The optimization result shows that (Figure 5.13), due to increased complexity in the target function, convergence becomes slower. However, the algorithm still manages to optimize the objective function, while almost monotonically increasing its value in the iterative process.

$$f(X) = \sum_{j=1}^{n} |(3 - 2x_i)x_i - x_{i-1} - x_{i+1} + 1|^p \tag{5.5}$$

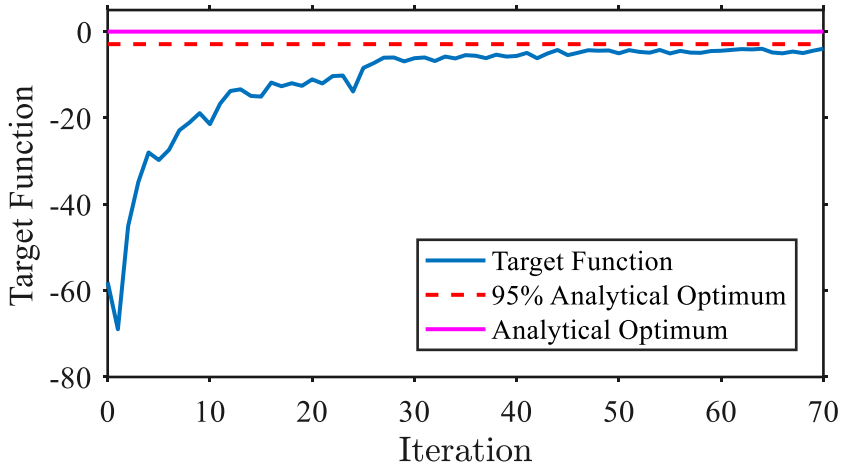

Figure 5.13 Broyden function optimization

5.6. Brown Function

Finally, the last optimization test was performed using the ten-variable version of the Brown function (5.6). As can be seen in Figure 5.14, BA algorithm is able to reach values close to 95% of the analytical optimum in 26 iterations.

$$f(X) = \sum_{i=1}^{k} (x_i - 1)^{x_i^2+1} + (x_i^2)^{x_{i-1}^2+1} \tag{5.6}$$

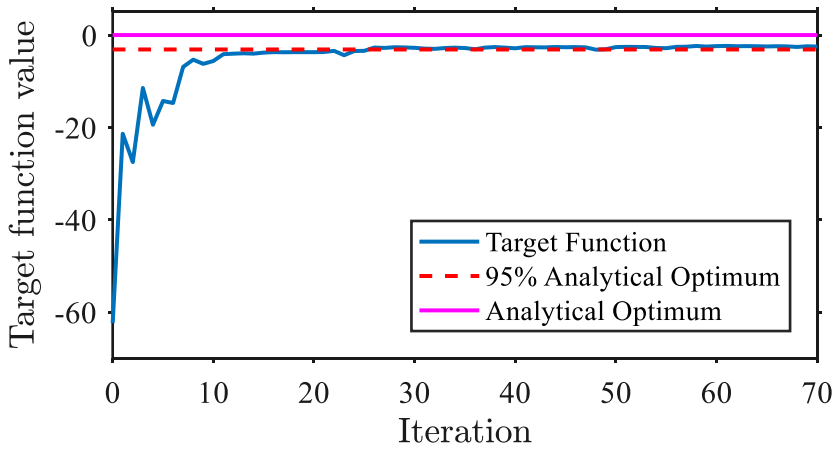

Figure 5.14 Brown function optimization

Chapter 6

Design of Trust Region Approach

As described in Chapter 4, BA algorithm improves traditional BO by imposing a trust region constraint in the sampling of new inputs. Such constraint restricts the search space to a region where the approximate model is assumed to faithfully represent the objective function.

The analytical function optimization tests, presented in the previous chapter, show that trust region initial sizing is a parameter with a key influence on the optimization procedure. Additionally, the dimension of the trust region is periodically adjusted at each iteration, following a certain update law.

Therefore, further exploration of this topic was conducted. In this chapter, the advantages and disadvantages of a trust region method are discussed, and four alternative approaches are presented.

6.1. Advantages and Disadvantages

The addition of a trust region constraint to an optimization problem offers several benefits that aim to improve the performance of the algorithm.

First of all, limiting the search space results in small and progressive changes to control actions. Such characteristic is particularly interesting when the optimization target is a physical system, which is the case in the wind farm optimization problem, since it is not desirable to drastically change control actions in consecutive inputs.

Secondly, a trust region approach also allows an almost monotonical increase in the objective function value. By constraining the sampling procedure, new inputs are selected from a region where the objective function approximation is assumed to be accurate. Said restriction should derive in a continuous growth of wind farm power, avoiding unexpected generation drops.

Lastly, trust region size is dynamically adjusted at each algorithm iteration depending on achieved target function improvement. On successful iterations, trust region size is expanded in order to expedite the convergence rate of the algorithm.

On the other hand, imposing a trust region constraint limits the scope of the search performed by the algorithm. As a result of said limitation, the optimization technique might convergence to a local optimum instead of the global target function optimum. In a wind farm power optimization scenario, this means that the optimized operation strategy might obtain suboptimal power production. In order to avoid local optimization, it is desirable that the algorithm is initialized close to the global maximum.

6.2. Alternative Trust Region Approaches

Aiming to improve BA algorithm performance, alternative trust region strategies were investigated. These alternative approaches include a modification of the original BA trust region, a bounded trust region approach and a self-adaptive trust region approach.

Performance of alternative trust region strategies was evaluated in the optimization of analytical test functions. In order to assimilate these optimization problems to a wind farm optimization environment, Gaussian uncertainty was added to every function output.

First of all, new approaches were tested on the exponential and trigonometric functions presented in the previous chapter (equations (5.2) and (5.3)). Following these tests, a second trigonometric function, identified as Trigonometric 2 and similar to the one presented in the previous chapter, was used. Finally, trust region strategies where also benchmarked in the optimization of a complex function with multiple local maximum and a single global maximum, named wavy function. Said function was obtained from a repository of global optimization benchmark problems[20].

Additionally, behaviour of alternative approaches was compared against the original BA trust region approach and a no-trust region approach.

$$f(X) = 1 - (x_1^2 \sin^2(x_1) + x_2^2 \sin^2(x_2)) \tag{6.1}$$

$$f(X) = 1 - \frac{1}{2}\left(\cos(10x_1)\, e^{-\frac{1}{2}x_1^2} + \cos(10x_2)\, e^{-\frac{1}{2}x_2^2}\right) \tag{6.2}$$

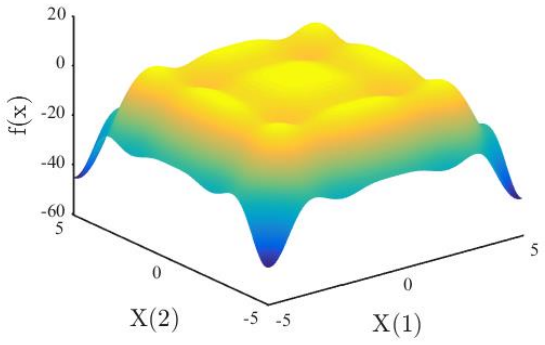

Figure 6.1 Trigonometric 2 function

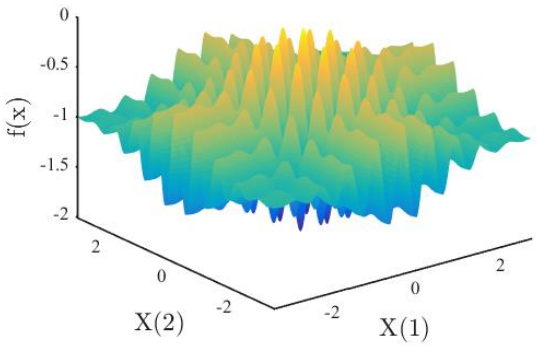

Figure 6.2 Wavy function

6.2.1. Modified Bayesian Ascent Trust Region

First of all, the first alternative trust region approach investigated is a modification of the original BA trust region update law. The originally proposed strategy is adjusted in order to ensure that the trust region only grows when a real improvement in the target function is observed. A real improvement is defined as an increase in the target function greater than the uncertainty present in the measurements. Therefore, the new trust region update law is defined as:

$$\begin{aligned} &if\ y^{n+1} - f^{max} \geq \max\left(\sigma_e\ ,\ \frac{\gamma}{n}(f^{max} - y^1)\right)\ then \\ &\qquad \tau^{n+1} = \beta\tau^n \\ &else \\ &\qquad \tau^{n+1} = \tau^1 \\ &end\ if \end{aligned} \tag{6.3}$$

$$\beta > 1 \tag{6.4}$$

Where y^{n+1}, f^{max}, n, γ and β are defined as described in 4.2, and σ_e is the uncertainty expected to exist in the target system measurements.

6.2.2. Bounded Trust Region

Secondly, a bounded trust region approach, that limits the maximum and minimum size of the trust region, was investigated. Such strategy aims to keep the algorithm search space within a range in which a trust region constraint has a positive effect on the optimization. The bounded trust region approach was proposed by Walmag [21] and is defined as:

$$\tau^{n+1} = \begin{cases} \max\left(\gamma_{dec}\tau^{n}, \tau^{\min}\right), & \rho < \eta_1 \\ \tau^{n}, & \eta_1 \leq \rho < \eta_2 \\ \min\left(\gamma_{inc}\tau^{n}, \tau^{\max}\right), & \eta_2 \leq \rho \end{cases} \tag{6.5}$$

$$\gamma_{inc} > 1, \qquad 0 < \gamma_{dec} < 1, \qquad 0 < \eta_1 \leq \eta_2 < 1 \tag{6.6}$$

Where γ_{dec} and γ_{inc} are the trust region decrease and increase coefficients, set to 0,9 and 1,1. The variables $\tau^{\min}$ and $\tau^{\max}$ set the maximum and minimum size of the trust region, set to 1% and 40% of the total search space. Finally, the parameter ρ controls the update law, and is defined as the ratio between the observed change in the target function and the improvement predicted by the approximation (6.7).

$$\rho = \frac{y^{n+1} - y^{n}}{\mu^{n+1} - \mu^{n}} \tag{6.7}$$

Where y is the observed target function value and μ is the predicted value by the optimization technique. If this ratio is negative or small, the iteration is considered unsuccessful since either the objective function decreased or it did increase, but not as much as predicted by the optimization. In either case, the size of the trust region is reduced. On the other hand, if the evaluated ratio is large an increase in the target function greater than predicted has been achieved, consequently, the trust region size is augmented. Finally, if the ratio has an intermediate value the trust region size is maintained. The ranges in which this ratio is considered big or small are controlled by the parameters η_1 and η_2, set to 0,01 and 0,95 respectively.

6.2.3. Self-Adaptive Trust Region

Finally, a self-adaptive trust region approach was investigated. This trust region update law was proposed by Walmag [21], and receives its name from the fact that the change in

trust region size is governed by a variable growth coefficient A. This strategy allows for fast growth when the observed improvement is similar to the estimation.

In order to modulate this trust region growth rate, the variable growth coefficient A is calculated at each iteration as a function of the parameter ρ, which is defined as described in the bounded trust approach (6.7). As shown in Figure 6.3, when the observed improvement matches the model prediction, i.e. ρ is close to one, the growth coefficient reaches its maximum value, set by the parameter α_2. On the other hand, inaccurate predictions result either in a trust region size decrease, if the target function has decreased, or in a smaller growth, if the improvement is larger than predicted.

Therefore, this latter approach addresses the issue with too successful iterations, in which the improvement in the target function is much larger than the one predicted by the model and the ratio ρ has a high value. Even though achieving a great increase in the objective function is desired, if such improvement is greater than what the model anticipated, it is an indication that the target function estimation is not faithful in the current search space. Therefore, increasing the trust region size would result in inaccurate predictions in future iterations.

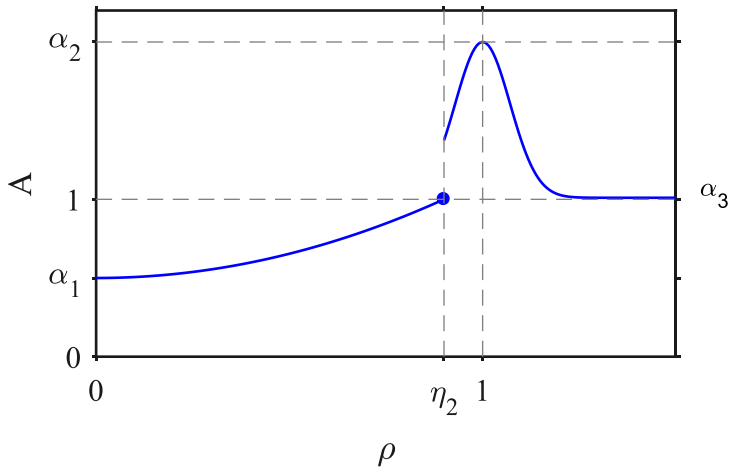

Figure 6.3 Self-adaptive trust region growth coefficient

The analytical expression for the described update law is formulated as:

$$\tau^{n+1} = A\tau^{n} \tag{6.8}$$

$$A = \begin{cases} \alpha_1, & \rho \leq 0 \\ \alpha_1 + (1-\alpha_1)\left(\frac{\rho}{\eta_2}\right)^2, & 0 < \rho \leq \eta_2 \\ \alpha_2 + (\alpha_2 - \alpha_3)\mathrm{e}^{-\left(\frac{\rho-1}{\eta_2-1}\right)^2}, & \eta_2 < \rho \end{cases} \quad (6.9)$$

$$\alpha_1 < 1 < \alpha_3 < \alpha_2, \qquad \eta_2 < 1 \quad (6.10)$$

Where α_1, $\alpha_2$and α_3 are the parameters that control the magnitude of the growth coefficient, and η_2 is the parameter that determines which analytical expression to use. The values of these parameters are set to 0,75, 2, 1,2 and 0,95 respectively.

6.3. Trust Region Approach Optimization Tests

Once the alternative trust region approaches had been implemented in the BA algorithm, they were tested in the optimization the four previously mentioned analytical functions: exponential, trigonometric, trigonometric 2 and wavy. These tests had the objective of determining the best performing trust region approach, which then was applied to the wind farm power maximization problem. This section presents the results of these optimizations tests, and justifies the selection of the trust region approach used in the in-operation learning technique.

First of all, the different trust region approaches were tested on the simplest function of the set, the exponential function. As was described in section 5.2, this function presents both a local and a global optimum in the search space. With the adequate trust region sizing, all trust region approaches manage to converge to the global maximum, as shown in Figure 6.4.

On the other hand, when neglecting the use of a trust region, the algorithm is unable to reach convergence in the 70 optimization iterations that were ran, as shown in Figure 6.5. As a consequence of the large search space, the sampling of new inputs is heavily focused in trying to learn more about the target function in order to obtain a better approximation. Therefore, the algorithm fails to locate the global optimum. However, given a large enough number of iterations, the algorithm would theoretically converge to the maximum.

This behaviour observed in the exponential function optimization when using no trust region, can also be observed when trying to optimize any of the other three test functions. Consequently, it was considered that a trust region approach was not suitable when trying to optimize a function in a low number of iterations.

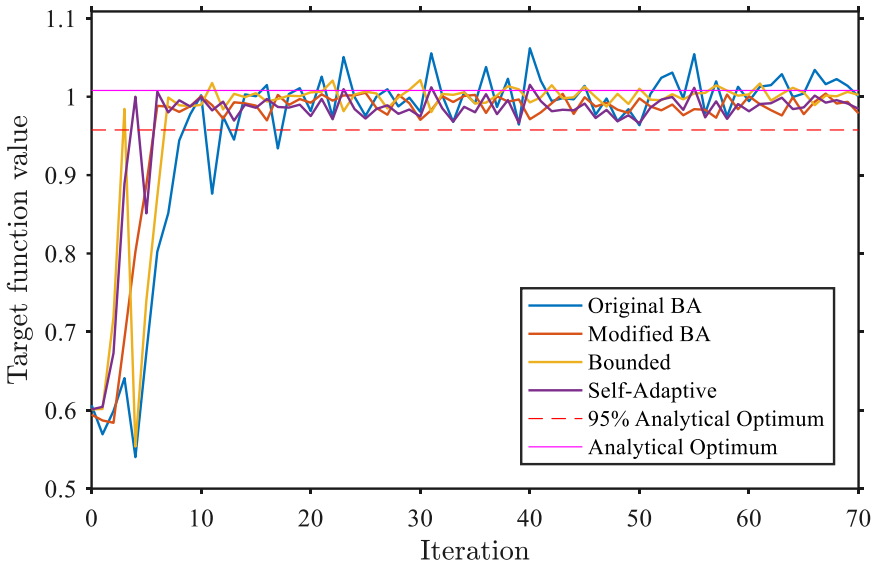

Figure 6.4 Exponential function optimization

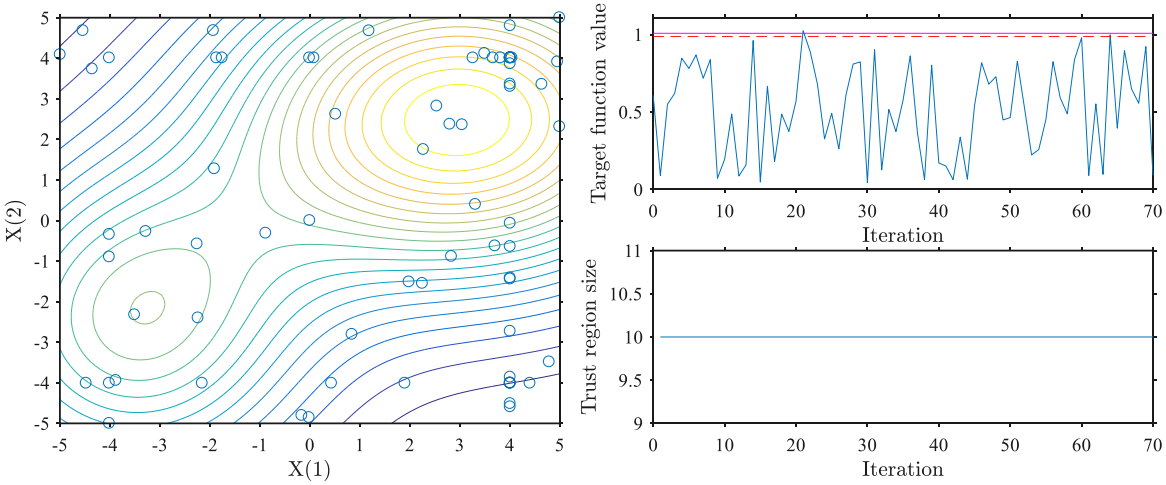

Figure 6.5 Exponential function optimization (No trust region)

Secondly, optimization tests on the trigonometric function presented in 5.3 were conducted. In this case, it was observed that the original BA, modified BA and bounded trust regions were only able to converge to a local optimum, as shown in Figure 6.6, when running the optimization with a small enough initial trust region. Running the optimization with a larger initial trust region, resulted in the algorithm being unable to converge to a solution.

On the other hand, the self-adaptive trust region approach proved to be able to find the global maximum of the function, located in the bounds of the search space. However, as can be seen in Figure 6.7, trust region size grew way over the bounds of the optimization. Since the optimization is already limited by the bounds on the variables, any increase in trust region size above the search space dimensions has no effect on the search space used by the algorithm. Therefore, it is effectively equivalent to running the optimization without a trust region. As a result of the way that the self-adaptive trust region is defined, when the trust region size becomes excessively large, a lot of unsuccessful iterations must be ran before it returns to normal values. For this reason, it is considered that adding a maximum trust region bound equal to the maximum search space would have a positive effect on this trust region approach, similar to adding a saturation value to an integer controller.

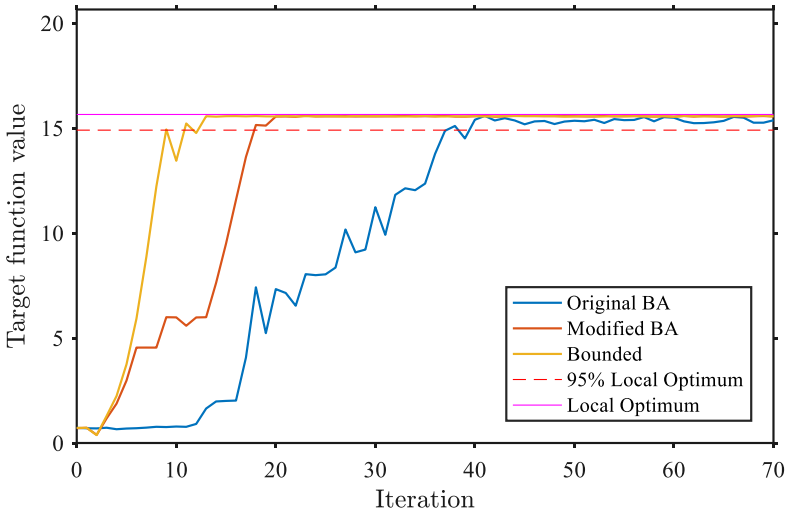

Figure 6.6 Trigonometric function optimization

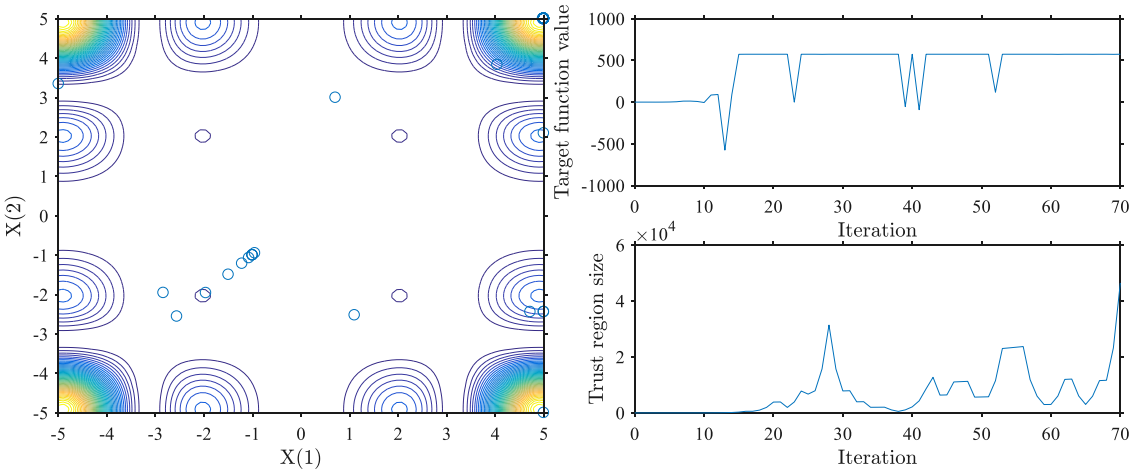

Figure 6.7 Trigonometric function Optimization (Self-Adaptive Trust Region)

Subsequently, optimization tests on the maximization of the trigonometric 2 function were conducted, defined in previously in this chapter. This function presents multiple local maximums in the chosen search space, which results in the algorithm being able to easily converge to one of them, as shown in Figure 6.8. However, in some of the optimization test ran, the bounded trust region approach had issues reliably converging to a solution, as can be seen in Figure 6.9. This undesired behaviour might be a negative of the addition of noise to the way the governing parameter ρ is calculated.

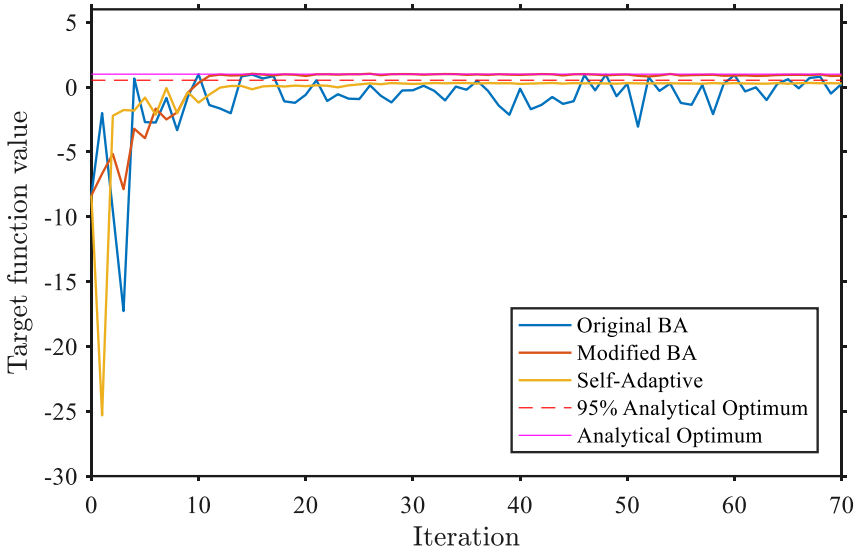

Figure 6.8 Trigonometric 2 optimization

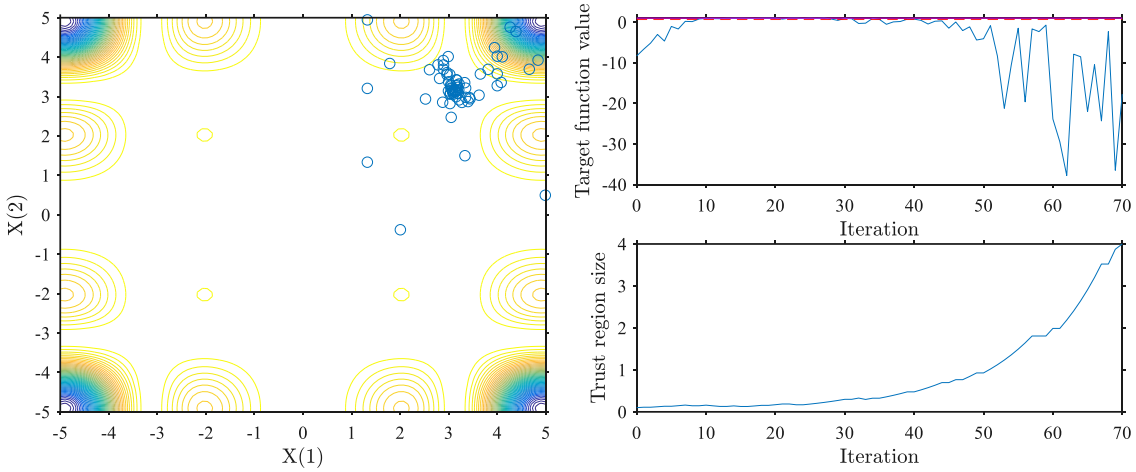

Figure 6.9 Trigonometric 2 function optimization Bounded trust region

Finally, the last optimization tests were conducted using the Wavy function, defined previously, as a target for the maximization. As a result of the complexity of this function, none of the trust region approaches where able to locate the global optimum. Convergence to local maximum was only observed when using a really small initial trust region size

(0,5% of the search space). In some simulations, the self-adaptive trust region approach was able to converge to higher target function values than other approaches. However, as can be seen in Figure 6.10, this was a result of a reduction of the trust region, therefore forcing convergence to a specific solution, and not the result of an actual successful optimization process. Such behaviour that locks the algorithm to a certain value is not desirable, especially when applying the optimization technique to the wind farm power maximization problem, since it is desirable to allow the algorithm a minimum size to keep exploring the target function.

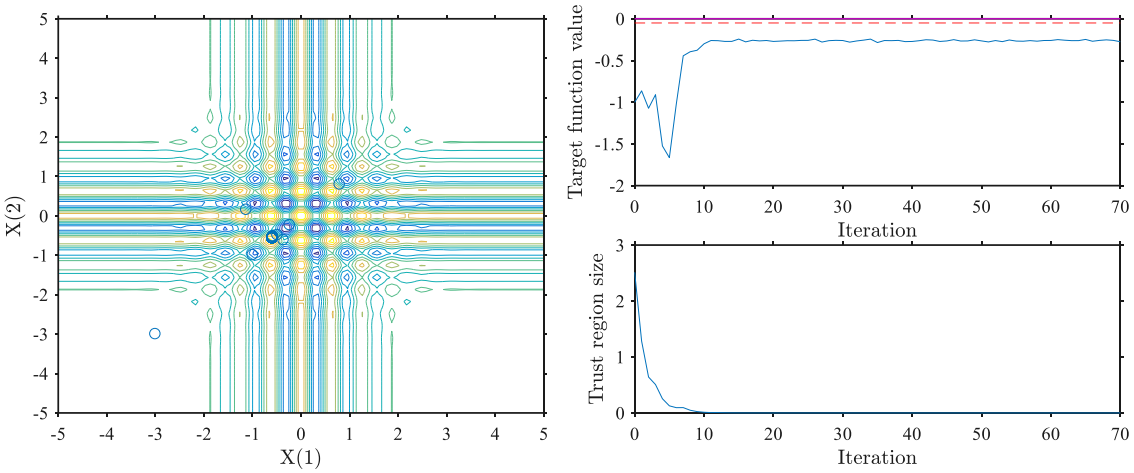

Figure 6.10 Wavy function optimization Self-Adaptive trust region

6.4. Choice of Trust Region Approach

As mentioned earlier, the objective of the optimization tests with the different trust region approaches was to determine the most adequate update law to be used in the wind farm power maximization problem. The conducted tests show that both the original BA and the modified BA approaches where performing the best. These update laws were able to optimize all three test functions successfully and without requiring any tuning of the parameters.

On the other hand, both the bounded trust region approach and the self-adaptive trust region approach showed undesired behaviour when trying to optimize some of the functions. It is possible that by adjusting some of the update law parameters, performance of these two approaches could be improved. Furthermore, the addition of noise to the target function might have an undesired effect on the way the parameter ρ governs trust region size.

Finally, as shown in the optimization test of the exponential function, using a no trust region approach is not a viable solution if the maximum has to be reached in a small number of iterations. The observed performance would only be worse when trying to apply this approach to the wind farm power maximization problem, given the increased complexity of the target system. Furthermore, when trying to control wind farm power production other requirements exist, that are not met by a no-trust region optimization. For instance, it is desirable that changes in control actions are performed gradually and that optimization is performed without drops in power production.

Accordingly to the observed results, it was considered that the best performing trust region update laws, when coupled with the BO algorithm and a target system with uncertainty, were the original BA and the modified BA approaches. Since no consistent difference in performance was observed, both these approaches were tested with the simulation tool presented in the next chapter, in order to see if the proposed modification can yield better performance in a more complex optimization environment.

Chapter 7

SimWindFarm Simulation of 3-Turbine Array

After implementation and validation of the BA algorithm optimization procedure, the developed in-operation learning technique is applied to the wind farm power maximization problem in the SWF [22] simulation environment. Using this tool, dynamic operation of a wind farm considering wind turbine dynamics, wind flow dynamics and wind farm controller dynamics can be simulated.

The aim of these simulation tests is to show that a coordinated power derating based strategy can actually improve wind farm power production, and to adjust the configuration of the implemented technique, in order to perform this optimization reliably. The resulting configuration and obtained wind farm power production improvement are presented in this chapter.

7.1. SimWindFarm Simulation Toolbox

SWF is a simulation toolbox that provides a wind farm MATLAB Simulink simulation environment for development of wind farm control algorithms. Using this tool, wind farm models with user defined turbine layouts can be generated, allowing for easy testing of different control strategies. SWF employs three control blocks to describe wind farm operation: wind field model, wind turbine model and wind farm controller model.

The wind field modelling can be divided into two different models: an ambient field model that describes the freestream wind flow, and a wake model that describes the wind speed deficits generated by turbine power extraction. In order to simplify the wind field

simulation models, constant mean wind speed and constant mean wind direction are assumed. Moreover, the wind field is simulated using a 2D grid in the plane at hub height and both ambient wind field model and wake model assume Taylor's frozen turbulence hypothesis for inviscid flow to be true [23]. As a result of this assumption, ambient wind field only needs to be generated in the most upstream line perpendicular to the wind direction. In order to initialize the wind field, the Kaimal spectrum is used. Detailed information on how this wind field is generated can be found in [22].

For wake effect calculations, SWF considers three effects: downwind wind speed reduction or wake deficit, wake region expansion, and wake meandering. Wake deficit and wake expansion are modelled using the Jensen wake model [8].

Regarding the wind turbine model, SWF assumes fixed turbine yaw angle and uses the NREL 5MW model to simulate turbine dynamics. Using this model, the pitch angle of the turbine is controlled in order to follow a derating factor set-point.

Finally, the wind farm controller model is in charge of setting the power demands of each turbine. The developed in-operation learning technique is embedded into this block, in order to determine this power set points following the optimized cooperative strategy.

7.2. Bayesian Algorithm Test Set-Up

In the interest of studying the performance the optimization technique in the simplest configuration with wake interaction effects, a three-turbine array was simulated in SWF (Figure 7.1).

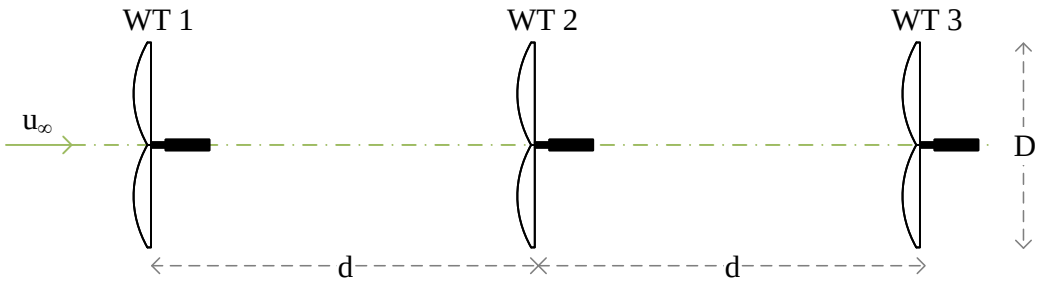

Figure 7.1 SWF turbine array

The simulated array is formed by three NREL 5MW turbines oriented perpendicular to the wind direction, which means that the yaw angle is set to 0º. This configuration, in which the turbines are aligned with the wind direction, is when effects of wake interaction

are most noticeable. Therefore, it is the setting that offers the highest potential for improvement when using a cooperative control approach. Average wind speed for the simulation is set below the rated wind speed value in order to keep the turbine operating in the desired power curve region, where the power normalization employed in the target function is valid. In order to have a large enough time interval to assess the performance of the in-operation learning technique, the length of the simulation for the optimization tests is set to 18.000 seconds. Full details on the parameters used in the simulation can be seen in Table 7.1.

Table 7.1 SWF simulation configuration

Parameter	Value
Average wind speed (u_∞)	8 m/s
Turbulence intensity	8 %
Average wind direction	0º
Wind turbine model	NREL 5MW
Rotor diameter (D)	126 m
Turbine spacing (d)	630 m
Rated power	5,3 MW
Rated wind speed	11,4 m/s
Optimization simulation time	18.000 s

As described in the formulation of the wind farm optimization problem in section 4.1, the variables used to control wind farm operation are the derating factors of the turbines in the array. In the conducted SWF simulation tests, this factors are defined as described in said section.

With the aim of evaluating the performance of the in-operation learning technique, optimized wind farm operation strategy was compared against both the greedy control strategy, where each turbine maximizes its own power production, and a model optimized solution obtained using the wind farm power function, derived from the Jensen wake model. Performance of the algorithm is assessed in terms of relative improvement, calculated as the relative increase in efficiency over the greedy strategy.

$$relative\ improvement\ (\%) = \frac{\eta_{BA,opt} - \eta_{greedy}}{\eta_{greedy}} \tag{7.1}$$

The optimized efficiency $\eta_{BA,opt}$ is defined as the average efficiency in the simulation after convergence has been reached. The rule used to determine convergence is based on the norm of the difference between two consecutive control action inputs, if the value of this norm stays below a threshold value of 0,001 during five successive iterations, it is assumed that the algorithm converged to a solution.

7.2.1. Model-Based Optimal Strategy

Wake models offer an approximation of the downstream wind field characteristics. By combining the approximation of the wake of all the turbines in the wind farm, an analytical wind farm power function can be estimated. Given that the operation strategy of the third turbine is fixed to its optimum value, a contour plot showing the resulting efficiency of a combined strategy for turbine one and turbine two can be obtained.

Using Jensen and Frandsen wake models the respective wind farm efficiency contour plots were obtained. As can be seen in Figure 7.2 and Figure 7.3, both plots show that the wind farm efficiency optimization is convex and presents a single global maximum. However, the models differ on the location of this optimum. According to the estimated efficiency from the Frandsen model, the optimal strategy for the three turbine setup is much closer to the greedy strategy, in which the derating factor is set to 1 for all turbines, than the one predicted by the Jensen model.

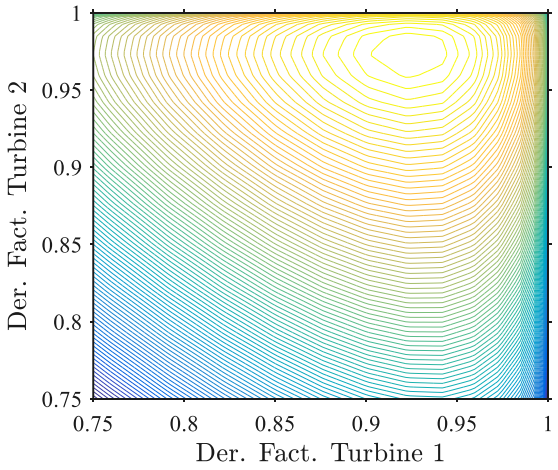

Figure 7.2 Jensen model efficiency plot

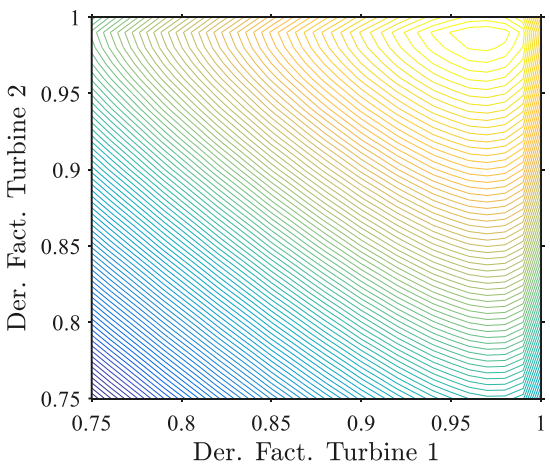

Figure 7.3 Frandsen model efficiency plot

When looking at the potential improvement that a cooperative control strategy can yield, in terms of relative improvement over the greedy strategy, the Jensen wake model predicts

a 3,99 % improvement when using the optimal control actions, whereas the Frandsen models lowers this value to 1,30%. Therefore, it can be assumed that the potential gain obtained by implementing coordinated control actions is quite small. However, it is expected that increasing the number of turbines in the array opens the possibility for higher potential gains.

7.2.2. Analysis of Objective Function Uncertainty

Normalizing the power production of the turbines by the incoming freestream wind speed has the effect of making the optimization independent from wind speeds variations. However, given the turbulent nature of the wind field, this also results in the introduction of additional noise in the target function.

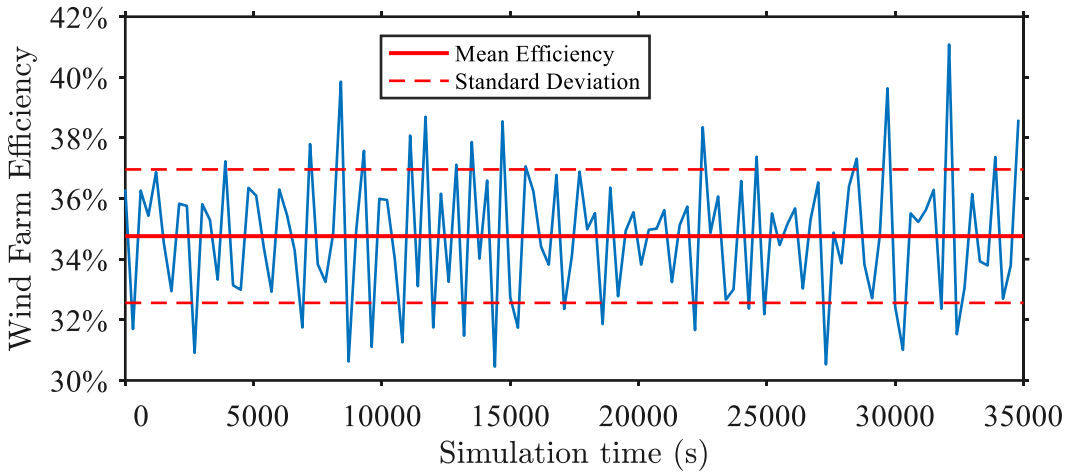

Figure 7.4 Greedy strategy simulation

As described in section 4.2, BA algorithm estimates the relationship between inputs and outputs of the system accounting for the uncertainty existing in the measurements. Therefore, computing an initial value for this uncertainty helps the hyperparameters optimization procedure. In order to estimate this uncertainty, two 36.000 seconds long simulations with constant control actions were run. In the first one, the greedy control strategy was maintained throughout the simulation, whereas on the second one, the optimized control actions from the Jensen model were used. In order to get faithful power and wind speed measurements, readings were time averaged in 300 seconds periods. The length of this interval was adjusted to allow enough time for the wind flow to stabilize after a change in operating conditions. As well as estimating the uncertainty existing in the measurements, these simulations also served to obtain the efficiency that the greedy control strategy yields, which is used to evaluate the performance of the algorithm.

The described setup resulted in the readings shown in Table 7.2. These results demonstrate that the efficiency obtained using a model-optimized strategy is only marginally higher than the one obtained by the non-cooperative approach. Figure 7.4 shows wind farm efficiency evolution throughout the simulation when maintaining the greedy strategy. As it can be seen in said figure, there is inconsistency in farm efficiency values that result in a high standard deviation value. Furthermore, this variability in wind farm efficiency is especially big when compared with the potential improvement estimated by Jensen and Frandsen wake models. According to these simulations, standard deviation of the efficiency is up to four times greater than the improvement predicted by the models.

Table 7.2 Constant control strategy simulations

Strategy	Derating Factor			Mean Efficiency (%)	Standard Deviation
	Turb. 1	Turb. 2	Turb. 3		
Greedy	1	1	1	34,7606	0,022014
Model Opt.	0,923	0,973	1	34,7626	0,016175

7.3. Delayed Implementation of Control Inputs and Measurement Post-Processing

The dynamic wind field model used in SWF not only accounts for the variance in incoming wind speed due to the turbulent nature of the flow, but the simulation also considers wind travel time. As a result of the data-driven approach of the algorithm, it is mandatory to make sure that output measurements are matched with the corresponding input. Therefore, it is necessary to contemplate these timings when computing wind turbine power measurements.

In order to cope with these difficulties, one strategy would be to increase the interval between optimization iterations, thus extending measurement averaging time. This interval should be long enough to allow for the wind to travel to the most downstream turbine, plus the averaging time for this last turbine measurements. Such implementation would limit the frequency in which the optimization can be executed. For instance, with the turbine spacing used in the simulation, and assuming the average wind speed value, 157,5 seconds are needed for wind to travel from the first to the last turbine. Since the in-

operation learning technique requires the power measurements from all the turbines in the array in order to calculate wind farm efficiency, and assuming a 300 seconds averaging time, this strategy would limit the periodicity of algorithm executions to 457,5 seconds. Moreover, the turbulent nature of the wind field results in varying wind travel time, for which the algorithm would need to account. As a result, algorithm execution periodicity would even need to be further increased.

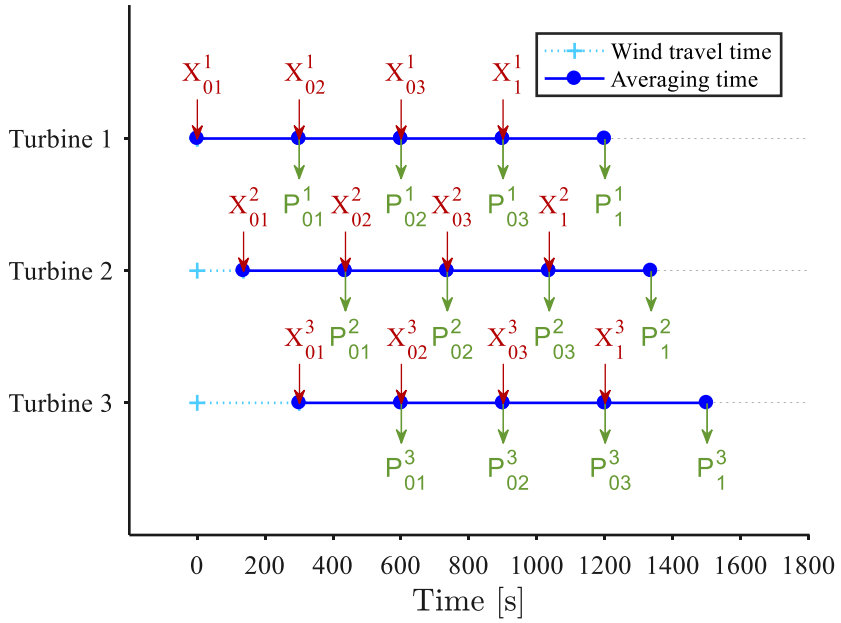

Figure 7.5 Delayed control action inputs

In order to reduce the time interval between subsequent algorithm iterations, a delayed implementation for control action inputs and measurement post-processing was built. Such implementation allowed the algorithm to change the operating strategy of a turbine without needing to wait for the wind to travel through all downstream turbines. Figure 7.5 shows a representation of the delayed implementation for control action inputs in the first four iterations. Following this strategy, the time between algorithm iterations is only limited by measurement averaging time. The resulting optimized control actions $x_i = (x_i^1, x_i^2, x_i^3)$ for each algorithm iteration i are set to the turbines accounting for wind travel times. Then turbine power measurements are output after the averaging interval. Since the power measurements from all wind turbines are needed before the optimization can be performed, a large enough set of buffered control action inputs x_0 is required in order to be able to control actions in initial iterations, when farm measurements are still not available. As shown in Figure 7.5, current array configuration requires 3 initial inputs for the delayed implementation to work. Once the fourth iteration is reached, turbine power

output measurements are post-processed in order to account for the time interval between the current iteration and the iteration in which the control input was set, as shown in Figure 7.6. At this fourth iteration, measurements from the first iteration control action are already available. Using this information, the first optimized control action x_1 is obtained and input to the turbines. After this point, turbine power outputs from previous iterations are used to find the optimized control actions.

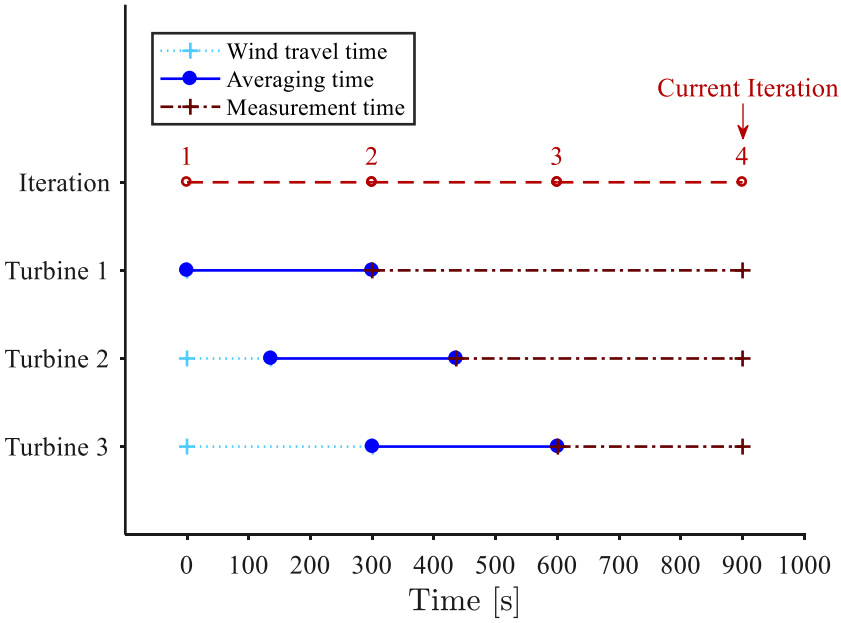

Figure 7.6 Delayed measurements post-processing

The developed delayed implementation results in a reduction in the time interval between algorithm iterations, therefore increasing the frequency in which the optimizer can be executed. Such strategy becomes especially interesting when the number of turbines in the array is large, since the time needed for wind to travel through all the farm is increased. On the other hand, this delayed implementation introduces wind travel time as another uncertainty factor in the measurements. This inaccuracy in wind travel time also limits the minimum averaging time that can be used. Consequently, it was considered that using 300 seconds as averaging time was a reasonable choice since it gives enough time for the wind flow to stabilize, and it is greater than the observed range of variations in wind travel time.

7.4. Bayesian Ascent Algorithm Configuration

After simulation environment settings were defined, BA algorithm configuration needed to be adjusted for the wind farm power maximization problem. In the interest of determining the most adequate settings for the in-operation learning technique, trial simulations were run to test the different configuration options.

First of all, algorithm initialization strategy needed to be defined. As described in the previous section, the delayed control action implementation requires a set of 3 initial inputs. A possible solution would be using randomized starting points. However, this resulted in the algorithm not being able to reliably improve wind farm efficiency over the greedy strategy in the fixed simulation time. As it was shown in the tests with analytical functions, using starting points closer to the optimum greatly improves the performance of the algorithm. For this reason, and after looking at the model calculated efficiency plots (Figure 7.2 and Figure 7.3), the initialization of the optimization was set to a combination of the greedy and the model optimized strategies. Since three starting sets of control actions were required, the third point was obtained as an average between the two strategies.

Secondly, using the information from the uncertainty analysis, an initial estimation of the uncertainty hyperparameter σ_e was set. Additionally, the initial trust region size τ_0 had to be specified. Moreover, following the tests of different trust region approaches in Chapter 6, a choice between two candidate trust region approaches original BA and modified BA, had to be made.

Lastly, another pending design choice was the initialization of the fmincon MATLAB function[24], used to maximize the EI when sampling new inputs. In this case two approaches were considered, using randomized initialization or the previous input with the highest observed output.

Given the number of parameters to adjust and the different values that these could take, a large number of simulations would need to be conducted to test all the possible configurations and possible interactions. Therefore, the design of experiments (DOE) method was used to reduce the amount of necessary tests.

7.4.1. Design of Experiments Method

The in-operation learning technique configuration problem involved setting four parameters: σ_e, τ_0, trust region approach and EI optimization initialization. The range of possible values for σ_e was estimated from the uncertainty analysis in 7.2.2. In the case of τ_0, this range was defined by the tests previously performed on analytical cases. In both cases, the design choice was simplified to four discrete values within the range. The possible levels for the four different parameters in the design problem are summarized in Table 7.3.

Table 7.3 Parameter levels

Parameter	Levels			
σ_e	0,010	0,015	0,020	0,025
τ_0	0,01	0,025	0,05	0,075
Trust region approach	Original BA (1)		Modified BA (2)	
EI opt. init.	Randomized (1)		Maximum (2)	

DOE is a systematic method for determining the relationship between parameters affecting a process and the observed system outputs. Using this information, the values of the parameters that yield the best output can be selected. Following the DOE method, experiments are designed to obtain the maximum information, consequently minimizing the number of necessary trial runs. Depending on the number of experiment parameters, and the possible values that these parameters can take, different DOE techniques can be used.

As a consequence of the time required for running SWF simulations, using a factorial DOE method is not a viable solution, since it involves testing all the possible combinations of parameters. Therefore, a method that reduced the number of necessary simulations was necessary. For this reason, the Taguchi DOE method was used. This method defines the set of experiments to be ran in order to find obtain a representative set that can be used to make design decisions. Further information on DOE and the different existing DOE methods can be found in [25]. Additionally, further information the Taguchi method can be found in [26].

Using the Taguchi method DOE implemented in Minitab Software, a set of 16 experiments with different algorithm configurations was defined (Table A.1). For each of the experiments in the set, 10 SWF simulations were conducted.

After gathering the data from the 16 experiments, optimization results were analysed using Minitab DOE Taguchi Design tool, in order to determine the optimal parameters values. Results of this analysis can be seen in Figure 7.7, where the vertical axis represents wind farm efficiency and the horizontal axis represents the previously defined levels for each one of the parameters. According to this analysis, the optimal values for σ_e and τ_0 are 0,015 and 0,025, respectively. Regarding the trust region approach, only a small difference optimized efficiency can be observed. Nevertheless, the original BA update manages to make the algorithm converge to a slightly higher efficiency value. Finally, looking at the EI optimization initialization results show that the best approach is to use the solution that yielded the maximum efficiency in previous iterations.

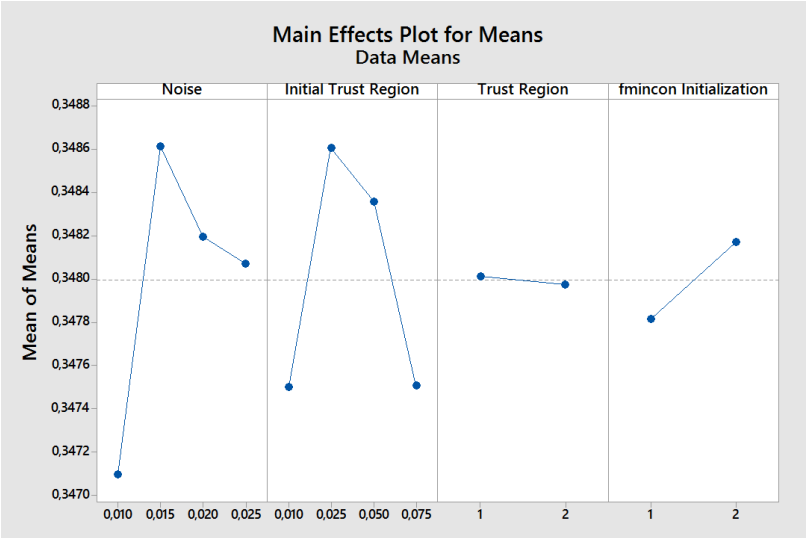

Figure 7.7 Analysis of Taguchi DOE

7.5. Results of Wind Farm Efficiency Optimization

After determining the best configuration for the in-operation learning tool, the wind farm efficiency optimization procedure was simulated using the selected parameters. Optimization results show that the algorithm can reliably improve wind farm efficiency, as can be seen in Figure 7.8. In all ten simulations, BA algorithm was able to converge to a solution that yielded higher efficiency than the non-cooperative control strategy, with an average improvement of 0,59 %.

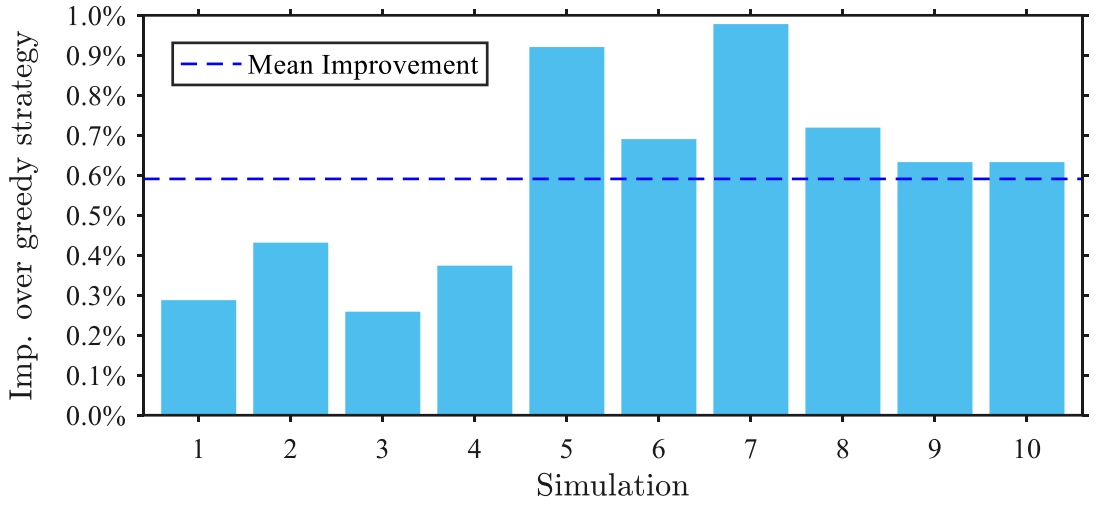

Figure 7.8 Wind farm efficiency improvement

Table 7.4 SWF simulation results

Sim.	Derating Factor			Conv. Ite	Conv. Time (min)	$\eta_{BA,opt}$	Rel. Imp.
	Turb. 1	Turb. 2	Turb. 3				
1	0,9911	0,9867	1	30	2,5	34,86 %	0,29 %
2	0,9881	0,9877	1	28	2,33	34,91 %	0,43 %
3	0,9868	0,9988	1	28	2,33	34,85 %	0,26 %
4	0,9869	0,9919	1	28	2,33	34,89 %	0,37 %
5	0,9874	0,9866	1	25	2,08	35,08 %	0,92 %
6	0,9861	0,9906	1	41	3,42	35,00 %	0,69 %
7	0,987	0,9868	1	37	3,08	35,10 %	0,98 %
8	0,9869	0,9957	1	25	2,08	35,01 %	0,72 %
9	0,992	0,9864	1	41	3,41	34,98 %	0,63 %
10	0,9913	0,9855	1	37	3,08	34,98 %	0,63 %
Mean	0,9883	0,9897	1	32	2,67	34,97 %	0,59 %
Std. Dev	0,0022	0,0045	0	6,34	0,53	0,00086	0,0025

The converging solutions of the different simulations can be seen in Table 7.4, as well as obtained efficiency and time needed to reach convergence. Optimized control actions are not consistent among the different simulation runs, even though the standard deviation is

small. Optimal control actions tend to be close to the greedy control actions, as predicted by the Frandsen wake model. Furthermore, the obtained relative improvement is also closer to the value calculated with the Frandsen model. Regarding the time needed for the algorithm to reach convergence, results show that on average 2,67 hours of wind farm operation are needed to optimize wind farm efficiency.

Finally, two sample plots of the optimization procedure are presented in Figure 7.9 and Figure 7.10. The first plot presents the evolution of the wind farm efficiency measurements, which presents a really big fluctuation, as predicted by the uncertainty analysis. On the other hand, the second plot shows the evolution of the wind turbine pitch angle, which the turbine variable controlled by the derating factor. This second plot shows that, even though uncertainty is present in wind farm efficiency values, the algorithm manages to converge to a solution that leads to stable pitch angle values for the turbines.

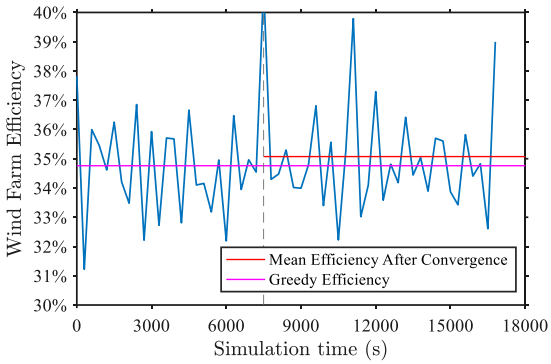

Figure 7.9 Wind farm efficiency evolution

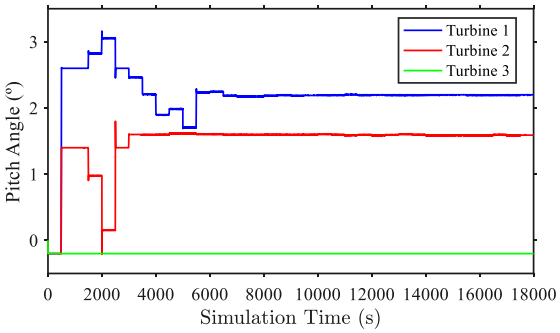

Figure 7.10 Pitch angle evolution

Chapter 8

Standalone Dynamic Wake Meandering Model-Based Simulation of 8-Turbine Array

In this chapter, the power production optimization of an 8-turbine array simulated using the sDWM model is presented. The sDWM model-based simulation environment offers an accurate representation of key wake dynamics, including turbine specific induction, build-up of wake turbulence, build-up of wake deficit, and influence of ambient turbulence intensity and atmospheric stability on wake effect . As in the previous chapter, the control approach for the optimization procedure is based on a power derating strategy.

Presented simulations aim to test the performance of the developed in-operation technique when optimizing the power production of an 8 turbine array, under conditions similar to real wind farm operation.

8.1. Standalone Dynamic Wake Meandering Model

The Dynamic Wake Meandering wind farm flow model was proposed by Larsen [27]. This model captures the most important features of wake dynamics and estimates the resulting power production and turbine loads. In order to conduct numerical simulations of the turbine power output in a less computationally intensive manner, a standalone implementation of said model was proposed by Keck et al. [28]. Using the standalone

approach, dynamically moving wake deficits and its time-averaged effect on turbine operation can be simulated.

In the DWM model, wake dynamics are separated into two processes: wake deficit evolution and wake meandering. The first process, wake deficit evolution, is governed by a steady-state thin-shear layer approximation of the Navier-Stokes equation proposed by Ainsle [29]. The wake meandering model, on the other hand, is based on the passive tracer assumption proposed by Larsen et al. [30]. Resulting estimations of wake deficit evolution and wake meandering are combined to find the time averaged flow field in the wind farm. Subsequently, the obtained flow field, combined with statistics on the dynamics of the wind farm, is used to estimate wind turbine power production. With the described approach, a simulation of farm power production can be performed while maintaining a low computational cost.

The main advantage of the sDWM model is its ability to model both wake deficit evolution and the increased turbulence in the wake. On the contrary, the wind farm flow model in SWF simulation environment assumed Taylor's frozen turbulence hypothesis, therefore disregarding the increased turbulence effect. Another this model is its capacity to account for the influence of ambient turbulence and atmospheric stability on wake effect.

On the other hand, the time-averaging procedure that the sDWM model employs results in the algorithm not being able to capture the uncertainty in power measurements observed in SWF simulations.

Validation tests on the sDWM model have shown that this model can successfully estimate wind farm power production. An average difference in the order of 5% was found when comparing the estimated power production against real wind farm data [31].

8.2. Bayesian Ascent Algorithm Test Set-Up

In order to evaluate the performance of the in-operation learning technique in an environment similar to real wind farm operation conditions, an 8-turbine array was simulated using the sDWM model-based simulation tool (Figure 8.1). This configuration was chosen because it represents a section of a typical offshore wind farm, thus simulating a real world optimization scenario.

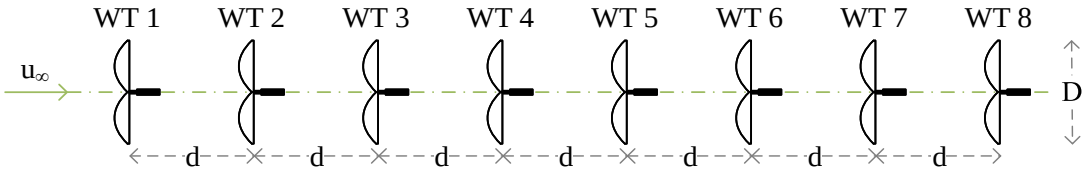

Figure 8.1 DWM turbine array

As in the SWF studied scenario, orientation of the eight turbines that form the array is set perpendicular to the incoming wind direction. Average wind speed for the simulation is set to 9 m/s, which is lower than the rated wind speed, therefore ensuring that the turbines operate in the desired turbine power curve region. Additionally, turbulence intensity of the turbulent wind field is set to 6 %. Finally, the length of the simulation is fixed at 70 optimization iterations which, assuming the 300 seconds averaging time, would correspond to a similar time frame as the one used in SWF. Full details on the parameters used in the simulation can be seen in Table 8.1.

Table 8.1 sDWM simulation configuration

Parameter	Value
Average wind speed (u_∞)	9 m/s
Turbulence intensity	6 %
Wind direction	0º
Rotor diameter (D)	93 m
Turbine spacing (d)	465 m
Optimization iterations	70

As described in section 4.1, control variables used for the optimization procedure are the derating factors of the turbines in the array. However, the way this parameter is defined in the sDWM model-based simulation tool differs from the previously used formulation. In this case, the derating factor of each turbine is defined as a fraction of the rated turbine power output:

$$DerFact_i = \frac{P_{d,i}}{P_{r,i}} \tag{8.1}$$

Where i is the turbine index, $P_{d,i}$ is the down-regulated power output of the turbine and $P_{r,i}$ is its rated power output.

The time-averaged nature of sDWM-simulated wind turbine operation means that for a certain control action input, power production measurements are deterministic. However, as SWF simulations showed, wind farm operation dynamics result in fluctuating efficiency values. The described uncertainty is not captured by sDWM simulations. Consequently, in order to obtain a better representation of real wind farm operation, Gaussian uncertainty is added to the efficiency measurements. The magnitude of said uncertainty was set to match the observed uncertainty in SWF simulations (section 7.2.2).

Two different optimization scenarios were analysed in sDWM simulations, with and without the described measurement uncertainty. For the first scenario, performance of the algorithm was analysed over 20 simulations, while for the second scenario the number of tests was reduced to 10. An overview of the optimization procedure with added uncertainty can be seen in Figure 8.2.

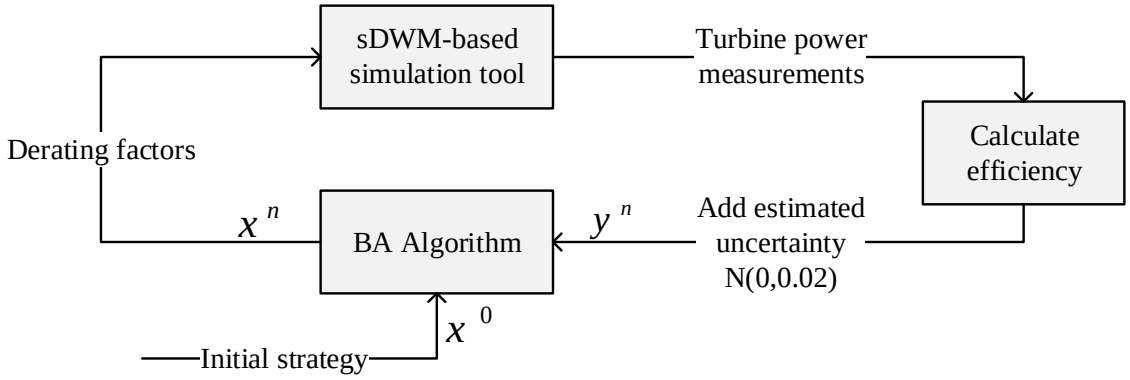

Figure 8.2 Optimization procedure

In the interest of evaluating the performance of the in-operation learning technique, the optimized wind farm operation strategy was compared against the greedy control strategy. As described in 7.2, optimized results are judged in terms of relative improvement over the greedy strategy. In order to further asses the optimization capabilities of the developed wind farm power maximizer, the resulting optimized efficiency is compared against that obtained when performing the maximization using SciPy Least-Squares (LS) optimizer [32].

8.3. Bayesian Ascent Algorithm Configuration

Regarding BA algorithm configuration for sDWM optimization simulation, both the trust region and the EI maximization initialization approaches were set to the optimal strategy

found in 7.4.1. However, as a result of the changes in simulation environment and length of the turbine array, the initial trust region size τ_0 value previously obtained was no longer the best performing configuration. For this reason, this parameter was adjusted by trial and error using the optimized value as a guideline. Consequentially, τ_0 was set to 0,01 in simulations with uncertainty, and to 0,075 in the no-uncertainty scenario. Finally, in simulations with added uncertainty, the initial estimation of σ_e was set to 0,02 (matching added uncertainty). On the other hand, in simulations without uncertainty, this value was set to 0,001.

Initial input strategy also differed between the two simulation scenarios. Even though setting a single initial input in the no-uncertainty scenario was enough, this was not the case when uncertainty was added. Therefore, the only initial input for the algorithm in the first scenario was the greedy control strategy. On the contrary, when uncertainty was added three initial inputs were needed to ensure that the algorithm performed and initial exploration of the search space.

8.4. Results of Wind Farm Efficiency Optimization

In this section results for the two test scenarios are presented, as well as the comparison between BA and LS optimization. Simulation tests show that the developed in-operation algorithm can reliably improve wind farm efficiency in both cases, while also outperforming the LS optimized solution.

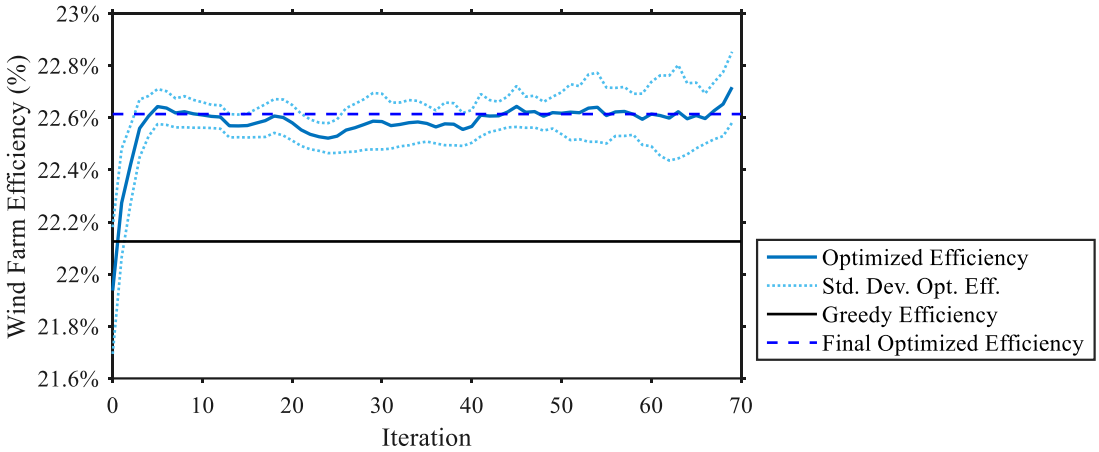

Figure 8.3 Optimized wind farm efficiency evolution (10 sim. av.)

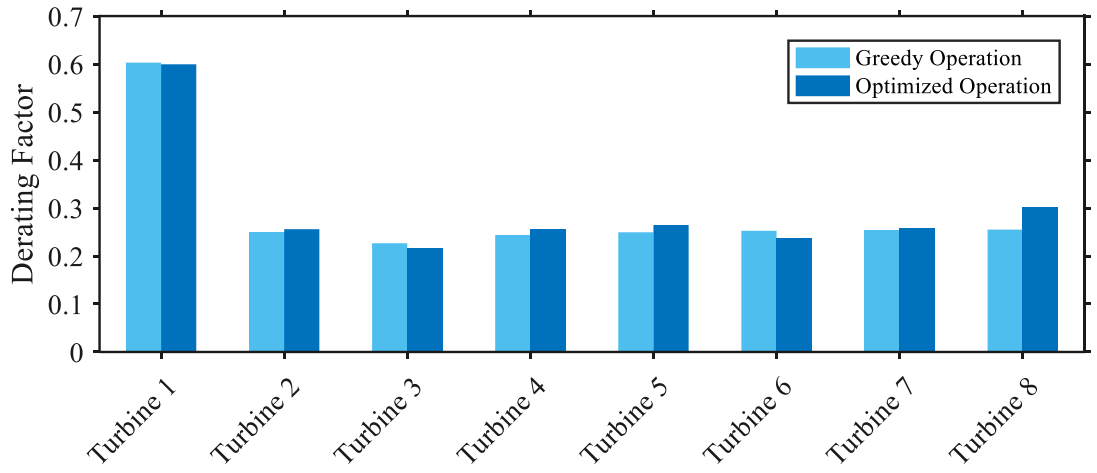

Figure 8.4 Optimized turbine set-points

First of all, the no-uncertainty scenario was analysed. Figure 8.3 shows the evolution of wind farm efficiency with BA optimization iterations. In spite of not having an uncertain system response, a certain variability exists between optimization runs due to the stochastic nature of the BA algorithm. The developed in-operation learning technique manages to improve greedy efficiency within the initial iterations of the optimization. Furthermore, wind farm efficiency rapidly converges to a stable efficiency value that improves conventional operation by 2,21 %. Towards the final simulation iterations, a growth trend in farm efficiency can be observed, which suggests potential for further operation enhancement. Over 10 simulation runs, the algorithm converged to two recurring solutions, which suggests that the in-operation learning technique operates consistently. In Figure 8.4 a comparison between optimized operation and conventional operation is shown. Generally, as a result of the optimization, upstream turbines are down-regulated, in order to allow for an increased power production in downstream turbines. Further details on optimization results are presented in Table B.1.

Secondly, the optimization capabilities of the in-operation learning technique with uncertainty added to the measurements were analysed. Figure 8.5 shows averaged wind farm efficiency evolution over 20 simulation runs. As a result of the added uncertainty, a much bigger fluctuation in efficiency values than in the previous scenario can be observed. The mean gain in wind farm efficiency when employing the optimized strategy is 1,42 %. The absolute value of this improvement is around 6 times smaller than the added uncertainty. Even with such a large fluctuation in efficiency values, Figure 8.6 shows that wind farm power production continuously stays above the value obtained with greedy operation. Even though no uncertainty was added to the displayed normalized

power production curve, uncertainty in the inputs to the algorithm, i.e. wind farm efficiency, results in variability of optimized control actions. This variability translates into the standard deviation shown in farm power production. However, as displayed in Figure 8.6, this variability is reduced as more optimization iterations are run, since the algorithm learns from the target system and adapts to the observed noise.

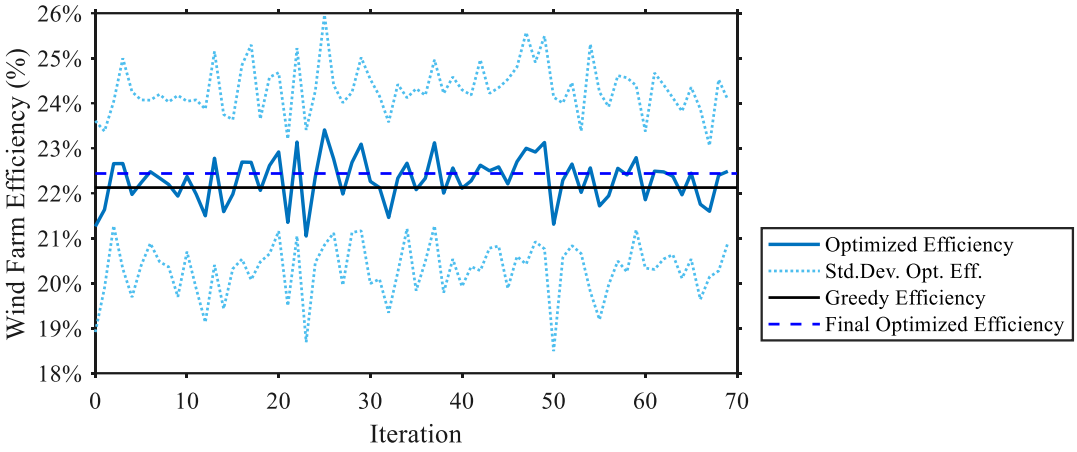

Figure 8.5 Optimized wind farm efficiency evolution (20 sim. av.)

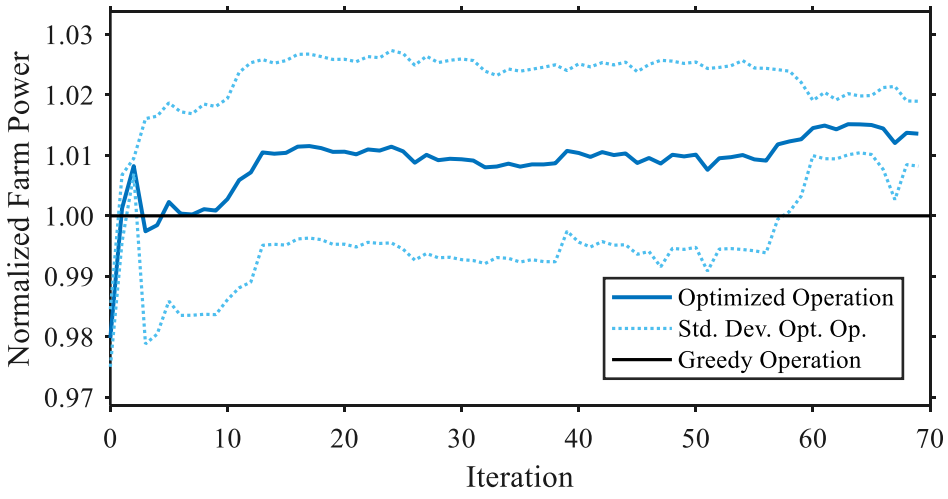

Figure 8.6 Wind farm power evolution (20 sim. av.)

In Figure 8.7 the variation in efficiency gain over the different optimization runs is shown. The change in efficiency gain in different runs is consequence of the variation in optimized operation strategy. As a result of the added uncertainty, the in-operation learning technique continuously changes the operating set points of the turbines. In this case, the optimized solution is considered as the averaged control action over the last 10 optimization iterations, since no convergence was observed when applying the previously defined criteria. Even though no recurring solution was observed, the average norm of

the difference between control action vectors (8.2) is only 0,033. Further details on optimization results are presented in Table B.2.

$$av\ norm = \frac{\sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N}|x_i - x_j|}{N(N-1)} \tag{8.2}$$

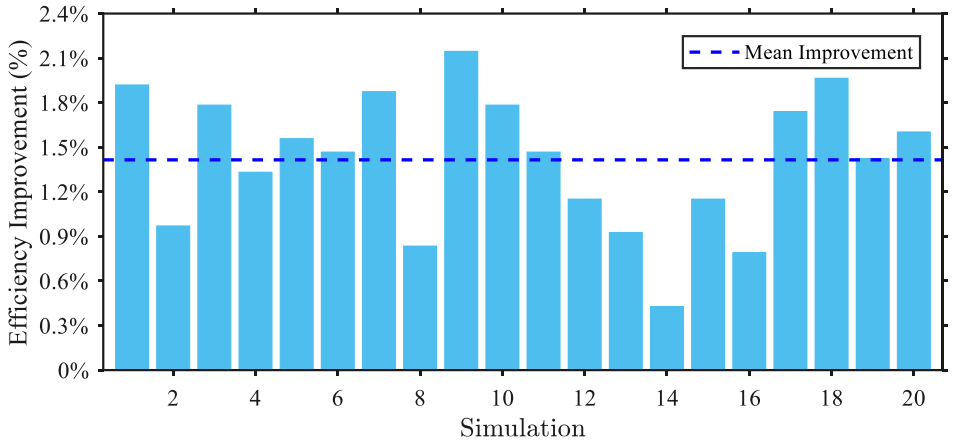

Figure 8.7 Efficiency improvement in different simulation runs

Finally, optimization performance of the in-operation learning technique is compared against LS optimization. In Figure 8.8, the mean and standard deviation of the optimized efficiency in the four studied cases are presented. As expected, a greater improvement in farm efficiency is observed when no uncertainty is added to farm measurements. Results show that BA algorithm outperforms LS optimization in both test scenarios. Furthermore, LS optimizer proved to be unable to reliably optimize wind farm efficiency when uncertainty was present in the measurements.

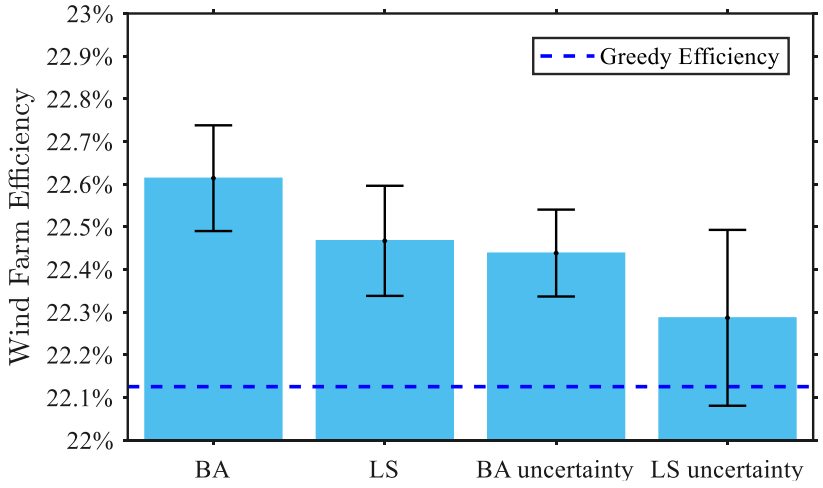

Figure 8.8 Comparison of BA and LS optimization

Chapter 9

Conclusions and Future Work

Implementation of a BA optimized control strategy that individually adjusts turbine power production set-points can reliably improve wind farm power output, without needing to construct and calibrate analytical farm operation models. In this work, BA algorithm was implemented in an in-operation learning and its performance was evaluated under conditions similar to real wind farm operation. This chapter first describes the conclusions derived from the present thesis, and then discusses future research directions.

9.1. Conclusions

In this thesis, an in-operation learning technique based on the BA algorithm was developed. Using said algorithm, wind power production can be maximized using only measured wind farm operational data, in a data-driven and model-free manner. In order to enhance this optimization procedure, alternative trust region update laws were explored. Additionally, a strategy for dealing with time-varying wind conditions was proposed and implemented. Furthermore, a delayed scheme for control action input and measurements post-processing was designed. Such scheme reduced the time interval between optimization iterations and dealt with the observed measurement uncertainty.

The developed in-operation learning technique was tested in two simulation tools that combined simulate wind farm operation dynamics relevant for wind farm control. First of all, the algorithm was tested in the SWF simulation tool, which covers wind flow, wind turbine and wind farm controller dynamics. In this simulation environment, wind farm

measurement uncertainty was estimated and the proposed delay scheme was validated. Secondly, simulations were conducted using a SDWM model-based simulation tool, which allows for the estimation of turbine power production while accounting for wake meandering effects. With the conducted tests in these two simulation tools, the derating-based optimization procedure was thus overall tested in conditions similar to real wind farms. BA algorithm reliably increased wind farm efficiency in all simulation scenarios, even with measurement uncertainty being six times greater than the potential optimization gain. Therefore, the results of this work suggest that BA algorithm can be applied to real wind farms. The characteristics of the developed technique make it suitable for wind farm power maximization, without requiring extensive on-site calibration like model-based approaches.

All in all, obtained results show that an induction-based cooperative control strategy can reduce the effects of wake interference, thus improving power production as compared to normal wind farm operation. Such enhancement of wind farm performance would translate into an improvement in wind farm economics for existing and future power plants, consequently contributing to further grow wind-energy based power generation.

9.2. Future Work

In order to enable the implementation of the developed in-operation learning technique to real wind farm operation, several research directions should be investigated. Particularly, performance under varying wind conditions, full-scale wind farm tests, combined induction-based and yaw misalignment control strategies, computational limitation when working with large data-sets and cost of energy formulation of the objective function.

First of all, the performance of the proposed in-operation learning technique, and the implemented strategy for dealing with time-varying environments should be tested in a dynamic wind condition simulation. The suggested intervals for the binning strategy might need to be adjusted in order to ensure that the optimization is valid in the whole interval.

Secondly, conducted simulation tests in this thesis have only been limited to turbine arrays, therefore a running a full-scale wind farm test would further investigate the performance of the BA algorithm. In order to do so, a clustering strategy that deals with

the aerodynamic interaction of different turbine arrays needs to be developed. This clustering strategy should be dynamically adjusted when changes in wind conditions are observed.

Additionally, the optimization capabilities of the algorithm when combining induction-based and yaw-misalignment-based control strategies should be investigated, since this last strategy has shown greater potential for wind farm power maximization. However, this combined strategy still needs to be tested in a dynamic simulation environment.

Usage of all historical data, which is one of the advantages of BO, conversely, also hinders the performance of the developed learning technique due to computation time limitations. Given a set of i training points, the dimensions of the covariance matrix K, which then has to be inverted when optimizing the hyperparameters, are $i\ x\ i$. Therefore, as the number of iterations grows, obtaining the new control action input becomes more and more computationally expensive. In order to deal with these issue, mathematical solutions that can simplify the calculations and a strategy that retains only the most relevant data-points should be investigated.

Finally, another possible line of research is the formulation of the objective function in terms of cost of energy, combining wind farm power production and turbine load levels, thus ensuring that the optimized solution does not degrade wind turbine life span.

Bibliography

[1] International Energy Agency, “IEA Wind 2015 Annual Report,” Paris, 226AD.

[2] B. Sanderse, “Aerodynamics of wind turbine wakes: Literature review,” *Energy Res. Cent. Netherlands*, pp. 1–46, 2009.

[3] J. R. Marden, S. D. Ruben, and L. Y. Pao, “A Model-Free Approach toWind Farm Control Using Game Theoretic Methods,” *Under Submiss.*, vol. 21, no. 4, pp. 1–11, 2012.

[4] P. M. O. Gebraad and J. W. Van Wingerden, “Maximum power-point tracking control for wind farms,” *Wind Energy*, vol. 18, no. 3, pp. 429–447, 2015.

[5] J. Park, “Data - Driven Cooperative Control for Wind Farm Power Maximization,” Stanford University, 2016.

[6] Y. Zhang, S. Wang, G. Ji, Y. Zhang, S. Wang, and G. Ji, “A Comprehensive Survey on Particle Swarm Optimization Algorithm and Its Applications,” *Math. Probl. Eng.*, vol. 2015, pp. 1–38, 2015.

[7] M. A. Ahmad, S. I. Azuma, and T. Sugie, “A model-free approach for maximizing power production of wind farm using multi-resolution Simultaneous Perturbation Stochastic Approximation,” *Energies*, vol. 7, no. 9, pp. 5624–5646, 2014.

[8] N. O. Jensen, “A note on wind generator interaction,” *Risø-M-2411 Risø Natl. Lab. Roskilde*, pp. 1–16, 1983.

[9] S. Frandsen, R. Barthelmie, S. Pryor, O. Rathmann, S. Larsen, and J. Hojstrup, “Analytical modelling of wind speed deficit in large offshore wind farms,” *Wind Energy*, vol. 9, no. 1–2, pp. 39–53, 2006.

[10] T. Knudsen, T. Bak, and M. Svenstrup, “Survey of wind farm control-power and fatigue optimization,” *Wind Energy*, vol. 18, no. August 2015, pp. 1333–1351, 2015.

[11] M. Mirzaei, G. Tuhfe, G. Giebel, P. E. Sørensen, and N. K. Poulsen, “Turbine

Control Strategies for Wind Farm Power Optimization," *Am. Control Conf.*, pp. 1709–1714, 2015.

[12] J. Park and K. H. Law, "A data-driven, cooperative wind farm control to maximize the total power production," *Appl. Energy*, vol. 165, no. January, pp. 151–165, 2016.

[13] J. Park and K. H. Law, "Bayesian Ascent: A Data-Driven Optimization Scheme for Real-Time Control with Application to Wind Farm Power Maximization," *IEEE Trans. Control Syst. Technol.*, vol. 24, no. 5, pp. 1655–1668, 2016.

[14] S. Zhong and X. Wang, "Decentralized Model-Free Wind Farm Control Via Discrete Adaptive Filtering Methods," *IEEE Trans. Smart Grid*, vol. 3053, no. c, pp. 1–1, 2016.

[15] M. R. Bonyadi and Z. Michalewicz, "Particle swarm optimization for single objective continuous space problems: a review," *Evol. Comput.*, no. x, p. EVCO_r_00180, 2016.

[16] B. Xing, *Intelligent Systems Reference Library 62 Innovative Computational Intelligence : A Rough Guide to 134 Clever Algorithms*. .

[17] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning.*, vol. 14, no. 2. 2004.

[18] J. Snoek, H. Larochelle, and R. Adams, "Practical Bayesian Optimization of Machine Learning Algorithms.," *Nips*, pp. 1–9, 2012.

[19] L. Lukˇ and J. Vlˇ, "Test Problems for Unconstrained Optimization san Test Problems for Unconstrained Optimization," no. 897, 2003.

[20] A. Gavana, "Global Optimization Benchmarks and AMPGO," 2013. [Online]. Available: http://infinity77.net/global_optimization/index.html. [Accessed: 05-May-2017].

[21] M. B. Walmag and J. M. Delhez, "A note on trust-region radius update ∗ ′," *Society*, vol. 16, no. 2, pp. 548–562, 2005.

[22] J. Grunnet, M. Soltani, and T. Knudsen, "Aeolus toolbox for dynamics wind farm model, simulation and control," *Eur. Wind Energy …*, p. 10, 2010.

[23] P. Davidson, *Turbulence: an introduction for scientist and engineers*. USA: Oxford University Press, 2015.

[24] C. Mathworks, "Optimization Toolbox: User 's Guide R 2017 a," 2017.

[25] G. Park, "Analytic Methods for Design Practice," *Anal. Methods Des. Pract.*, pp. 309–391, 2007.

[26] G. Taguchi, *System of experimental design: engineering methods to optimize quality and minimize costs*, no. v. 1. UNIPUB/Kraus International Publications, 1987.

[27] G. C. Larsen, *Dynamic wake meandering modeling*, vol. 1607, no. June. 2007.

[28] R.-E. Keck, M. de Maré, M. J. Churchfield, S. Lee, G. Larsen, and H. A. Madsen, "Two improvements to the dynamic wake meandering model: including the effects of atmospheric shear on wake turbulence and incorporating turbulence build-up in a row of wind turbines," *Wind Energy*, vol. 18, no. 1, pp. 111–132, 2015.

[29] J. F. Ainslie, "Calculating the flowfield in the wake of wind turbines," *J. Wind Eng. Ind. Aerodyn.*, vol. 27, no. 1, pp. 213–224, 1988.

[30] G. C. Larsen, H. A. Madsen, K. Thomsen, and T. J. Larsen, "Wake meandering: a pragmatic approach," *Wind Energy*, vol. 11, no. 4, pp. 377–395, 2008.

[31] R.-E. H. J. Keck, "Validation of the standalone implementation of the dynamic wake meandering model for power production," *Wind Energy*, vol. 18, no. 9, pp. 1579–1591, 2015.

[32] SciPy Community, "SciPy Reference Guide Release 0.16.0," p. 1229, 2015.

Appendix A:

Design of Experiments Simulations

Table A.1 Design of Experiments simulations

Experiment	σ_e	τ_0	Trust region approach	EI opt. init.
1	0,01	0,01	Original BA	Maximum
2	0,01	0,025	Original BA	Maximum
3	0,01	0,05	Modified BA	Randomized
4	0,01	0,075	Modified BA	Randomized
5	0,015	0,01	Original BA	Randomized
6	0,015	0,025	Original BA	Randomized
7	0,015	0,05	Modified BA	Maximum
8	0,015	0,075	Modified BA	Maximum
9	0,02	0,01	Modified BA	Maximum
10	0,02	0,025	Modified BA	Maximum
11	0,02	0,05	Original BA	Randomized
12	0,02	0,075	Original BA	Randomized
13	0,025	0,01	Modified BA	Randomized
14	0,025	0,025	Modified BA	Randomized

15	0,025	0,05	Original BA	Maximum
16	0,025	0,075	Original BA	Maximum

Appendix B:

Standalone Dynamic Wake Meandering Model Optimization Results

B.1. Bayesian Ascent Optimization Without Uncertainty

Table B.1 BA optimization without uncertainty results

Sim.	Derating Factors								$\eta_{BA,opt}$	η_{greedy}	Rel. Imp.
	Turb. 1	Turb. 2	Turb. 3	Turb. 4	Turb. 5	Turb. 6	Turb. 7	Turb. 8			
1	0,5944	0,2671	0,2369	0,2417	0,2403	0,2442	0,2436	1	22,47%	22,13%	1,56%
2	0,613	0,2707	0,2055	0,2661	0,288	0,2305	0,2649	1	22,71%	22,13%	2,64%
3	0,5944	0,2671	0,2369	0,2417	0,2403	0,2442	0,2436	1	22,47%	22,13%	1,56%

4	0,613	0,2707	0,2055	0,2661	0,288	0,2305	0,2649	1	22,71%	22,13%	2,64%
5	0,613	0,2707	0,2055	0,2661	0,288	0,2305	0,2649	1	22,71%	22,13%	2,64%
6	0,613	0,2707	0,2055	0,2661	0,288	0,2305	0,2649	1	22,71%	22,13%	2,64%
7	0,613	0,2707	0,2055	0,2661	0,288	0,2305	0,2649	1	22,71%	22,13%	2,64%
8	0,5944	0,2671	0,2369	0,2417	0,2403	0,2442	0,2436	1	22,47%	22,13%	1,56%
9	0,5944	0,2671	0,2369	0,2417	0,2403	0,2442	0,2436	1	22,47%	22,13%	1,56%
10	0,613	0,2707	0,2055	0,2661	0,288	0,2305	0,2649	1	22,71%	22,13%	2,64%
Mean	0,6056	0,2693	0,2181	0,2563	0,2689	0,2360	0,2564	1	22,64%		2,21%
SDev	0,0096	0,0019	0,0162	0,0126	0,0246	0,0071	0,0110	0	0,0012		0,005

B.2. Bayesian Ascent Optimization With Uncertainty

Table B.2 BA optimization with uncertainty results

Sim.	Derating Factors								$\eta_{BA,opt}$	η_{greedy}	Rel. Imp.
	Turb. 1	Turb. 2	Turb. 3	Turb. 4	Turb. 5	Turb. 6	Turb. 7	Turb. 8			
1	0,6003	0,2492	0,2266	0,2266	0,2437	0,2508	0,2562	1	22,55%	22,13%	1,92%
2	0,5975	0,2482	0,2254	0,237	0,2513	0,2545	0,25	1	22,34%	22,13%	0,97%
3	0,6063	0,2539	0,229	0,2419	0,2479	0,251	0,2552	1	22,52%	22,13%	1,78%

4	0,6022	0,2507	0,2276	0,237	0,2504	0,2539	0,2499	1	22,42%	22,13%	1,33%
5	0,605	0,2766	0,2718	0,2347	0,2477	0,2528	0,2534	1	22,47%	22,13%	1,56%
6	0,6095	0,2358	0,2322	0,2386	0,246	0,2468	0,2399	1	22,45%	22,13%	1,47%
7	0,6033	0,2757	0,2267	0,2444	0,2486	0,2524	0,2512	1	22,54%	22,13%	1,87%
8	0,6073	0,2625	0,2346	0,2279	0,2439	0,2473	0,239	1	22,31%	22,13%	0,83%
9	0,6086	0,2499	0,2264	0,2437	0,2486	0,2528	0,2654	1	22,60%	22,13%	2,15%
10	0,6035	0,2501	0,2262	0,2419	0,25	0,2519	0,241	1	22,52%	22,13%	1,78%
11	0,6034	0,2484	0,2251	0,2429	0,2492	0,2529	0,2474	1	22,45%	22,13%	1,47%
12	0,6041	0,2518	0,2672	0,2405	0,2359	0,252	0,2468	1	22,38%	22,13%	1,15%
13	0,5715	0,2358	0,2147	0,258	0,2594	0,2666	0,269	1	22,33%	22,13%	0,92%
14	0,6045	0,2514	0,2271	0,225	0,2452	0,2527	0,2173	1	22,22%	22,13%	0,43%
15	0,6041	0,2518	0,2672	0,2405	0,2359	0,252	0,2468	1	22,38%	22,13%	1,15%
16	0,6047	0,2593	0,2352	0,2312	0,2372	0,2508	0,2399	1	22,30%	22,13%	0,79%
17	0,6042	0,2506	0,2273	0,2447	0,2486	0,2532	0,2405	1	22,51%	22,13%	1,74%
18	0,6239	0,2498	0,2343	0,2423	0,2479	0,251	0,2549	1	22,56%	22,13%	1,96%
19	0,598	0,254	0,2253	0,2402	0,2519	0,2551	0,257	1	22,44%	22,13%	1,42%
20	0,6059	0,2523	0,2278	0,2263	0,2473	0,2524	0,2355	1	22,48%	22,13%	1,60%

Mean	0,6034	0,2529	0,2339	0,2383	0,2468	0,2526	0,2478	1	22,44%		1,42%
SDev	0,0092	0,0100	0,0157	0,0080	0,0056	0,0039	0,0114	0	0,001		0,0046

A.1. Least-Squares Optimization Without Uncertainty

Table B.3 LS optimization without uncertainty results

Sim.	Derating Factors								$\eta_{BA,opt}$	η_{greedy}	Rel. Imp.
	Turb. 1	Turb. 2	Turb. 3	Turb. 4	Turb. 5	Turb. 6	Turb. 7	Turb. 8			
1	0,5969	0,2500	0,2252	0,2442	0,2506	0,2560	0,2523	1	22,49%	22,13%	1,66%
2	0,6009	0,2501	0,2209	0,2455	0,2509	0,2535	0,2550	1	22,39%	22,13%	1,18%
3	0,6030	0,2499	0,2266	0,2438	0,2494	0,2526	0,2539	1	22,54%	22,13%	1,89%
4	0,6026	0,2499	0,2264	0,2439	0,2496	0,2524	0,2538	1	22,54%	22,13%	1,87%
5	0,6036	0,2499	0,2263	0,2436	0,2497	0,2528	0,2542	1	22,58%	22,13%	2,05%
6	0,7839	0,2105	0,2945	0,3169	0,3242	0,3284	0,2322	1	22,26%	22,13%	0,61%
7	0,6028	0,2499	0,2265	0,2437	0,2493	0,2525	0,2539	1	22,53%	22,13%	1,85%
8	0,6995	0,2234	0,2945	0,3169	0,1844	0,3283	0,3300	1	22,23%	22,13%	0,47%
9	0,6028	0,2499	0,2266	0,2438	0,2495	0,2526	0,2539	1	22,54%	22,13%	1,88%
10	0,6031	0,2499	0,2274	0,2447	0,2485	0,2535	0,2530	1	22,57%	22,13%	1,99%

Mean	0,6299	0,2433	0,2395	0,2587	0,2506	0,2683	0,2592	1	22,47%		1,54%
SDev	0,0622	0,0143	0,0291	0,0307	0,0330	0,0317	0,0258	0	0,0013		0,006

A.2. Least-Squares Optimization With Uncertainty

Table B.4 LS optimization with uncertainty results

Sim.	Derating Factors								$\eta_{BA,opt}$	η_{greedy}	Rel. Imp.
	Turb. 1	Turb. 2	Turb. 3	Turb. 4	Turb. 5	Turb. 6	Turb. 7	Turb. 8			
1	0,6029	0,2500	0,2266	0,2438	0,2494	0,2526	0,2539	1	22,57%	22,13%	2,01%
2	0,6029	0,2500	0,2266	0,2438	0,2494	0,2526	0,2539	1	22,13%	22,13%	0,00%
3	0,6029	0,2499	0,2266	0,2438	0,2494	0,2574	0,2539	1	22,57%	22,13%	2,00%
4	0,6030	0,2499	0,2266	0,2438	0,2484	0,2526	0,2539	1	22,54%	22,13%	1,88%
5	0,6029	0,2500	0,2266	0,2438	0,2494	0,2526	0,2539	1	22,13%	22,13%	0,00%
6	0,6030	0,2450	0,2266	0,2438	0,2494	0,2526	0,2539	1	22,56%	22,13%	1,96%
7	0,6029	0,2500	0,2266	0,2438	0,2494	0,2526	0,2539	1	22,13%	22,13%	0,00%
8	0,6029	0,2500	0,2266	0,2438	0,2494	0,2526	0,2539	1	22,13%	22,13%	0,00%
9	0,6029	0,2499	0,2266	0,2438	0,2494	0,2526	0,2539	1	22,41%	22,13%	1,27%
10	0,6030	0,2499	0,2266	0,2438	0,2494	0,2526	0,2539	1	22,46%	22,13%	1,51%

11	0,6029	0,2499	0,2266	0,2438	0,2494	0,2526	0,2539	1	22,13%	22,13%	0,00%
12	0,6029	0,2499	0,2266	0,2438	0,2494	0,2526	0,2539	1	22,13%	22,13%	0,00%
13	0,6030	0,2499	0,2266	0,2438	0,2494	0,2526	0,2539	1	22,55%	22,13%	1,94%
14	0,6029	0,2499	0,2266	0,2438	0,2494	0,2526	0,2539	1	22,13%	22,13%	0,00%
15	0,6029	0,2499	0,2266	0,2438	0,2494	0,2526	0,2539	1	22,57%	22,13%	2,01%
16	0,6030	0,2499	0,2266	0,2438	0,2494	0,2526	0,2539	1	22,13%	22,13%	0,00%
17	0,6030	0,2499	0,2266	0,2438	0,2494	0,2526	0,2539	1	22,13%	22,13%	0,00%
18	0,6029	0,2499	0,2266	0,2438	0,2494	0,2526	0,2539	1	22,13%	22,13%	0,00%
19	0,6029	0,2499	0,2266	0,2438	0,2494	0,2526	0,2539	1	22,13%	22,13%	0,00%
20	0,6030	0,2499	0,2266	0,2438	0,2494	0,2526	0,2539	1	22,13%	22,13%	0,00%
Mean	0,6030	0,2497	0,2266	0,2438	0,2494	0,2528	0,2539	1	22,29%		0,73%
SDev	0,0000	0,0011	0,0000	0,0000	0,0002	0,0011	0,0000	0	0,0021		0,009