# Dynamic Wind Farm Control using the WFSim flow model

J.A. Frederik

Master of Science Thesis

TU Delft Delft University of Technology

Delft Center for Systems and Control

# Dynamic Wind Farm Control using the WFSim flow model

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

J.A. Frederik

February 2, 2017

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

The cover photo is of the Princess Amalia Wind Farm in the Dutch North Sea, under the Courtesy of *Rijkswaterstaat*.


# Abstract

Renewable energy is becoming more and more important in today's society. Wind energy plays an important role in the production of renewable energy. Due to economic advantages, wind turbines are often sited close together, creating wind farms. As a result, the wind turbines in the farm become interconnected due to the wakes of the turbines. In conventional wind farm control, this interconnection is ignored, and wind turbines are operated at their own local optimum. This control strategy is referred to as greedy control.

In the wake of a turbine, the wind speed is reduced. As a result, the power production of a wind turbine that is situated in a wake is also significantly lower. The greedy control strategy may therefore not be optimal for wind farm control, which subsequently has been receiving an increasing amount of attention recently. Most research on wind farm control focusses on steady-state optimization, *i.e.*, finding the optimal steady-state given certain constant conditions. Since wind flow is always subject to change, the potential gain of this procedure is limited.

In this thesis, a closed-loop dynamic wind farm controller is presented using a Model Predictive Control (MPC) framework. The control objective is to maximize the power production of the wind farm, resulting in an economic MPC problem. As the optimal input is time-dependent, the number of control inputs and states involved increase as the prediction horizon of the MPC problem increases, resulting in a more complex problem than the steady-state optimization problem.

As a model, Wind Farm Simulator (WFSim) is used, which is based on 2-dimensional Navier-Stokes equations. This results in a high-dimensional problem from which the optimum is unknown. The adjoint method is applied to determine the gradient of the objective function in a time-efficient manner. The controller developed in this thesis increases the power production of a wind farm with respect to the conventional greedy control stategy, and is able to adapt to changes in the atmospheric conditions. The adjoint-based MPC algorithm therefore shows real potential to perform real-time dynamic control on wind farms.

# Table of Contents

# List of Figures

# List of Tables

# Acknowledgements

I would like to thank my supervisor dr.ir. J.W. van Wingerden for his assistance, guidance and motivation during the work on this thesis. Without his help I would probably still be running fruitless simulations in MATLAB. I would also like to thank my daily supervisor ir. S. Boersma who has spent countless hours assessing this thesis and providing some very useful feedback. Thanks also go out readers ir. B.M. Doekemeijer and S.Y. Deen for providing feedback on this thesis in order to make sure that it is also understandable for people not involved in the project.

Furthermore, I would of course like to thank my parents, Hans and Dorothé, for raising me to become the man that is now on the verge of earning his ir.-title. Without their moral (and financial) support, I would never have been able to come this far. I would also like to thank my girlfriend, Saimi, for coping with me when I'd had a bad day, and for consistently waiting for me with dinner when I was working late on finishing this thesis these last few weeks.

Moreover, I would like to thank all my friends and relatives that have been with me all this time, and for whom I've had very little time over the last few months. When I look back on my time at the Delft University of Technology (DUT), the fun I've had with you is probably what I will remember most.

Finally, I would like to thank the reader for taking the time to study what I have been working on for the last year-and-a-bit. If you will learn only a fraction of what I have learned from it, I will consider this thesis a success.

Thank you all.

Delft, University of Technology
February 2, 2017

J.A. Frederik

# Chapter 1

# Introduction

In this chapter, the topics concerning this thesis will be introduced. The motivation for the research done in this thesis will be presented in Section 1-1, followed by an overview of state-of-the-art in the field of wind farm modeling and control. Section 1-3 will then present the problem statement, as well as the approach followed in this thesis. Finally, Section 1-4 presents the structure of this report.

## 1-1 Motivation

Renewable energy is an ever increasing branch in technology, due to concerns about the environment. Former US President and Nobel Peace Prize winner Barack Obama has stated that he believes the trend towards clean energy is irrevertible [1]. Obama is not the only (former) world leader who feels that the transition towards renewable energy is essential. On the 2015 UN Climate Change Conference, all 195 participating countries agreed to reduce the carbon dioxide emission in order to keep the global warming below 2 degree Celsius [2]. To achieve this, a big role is reserved for wind energy. The vast majority of wind energy produced wordwide is generated in wind farms. A wind farm is a (large) number of wind turbines that are grouped together. Wind farms have several advantages compared to individual wind turbines [3]:

- The cost of installation and maintenance is reduced;
- It is easier and cheaper to connect the wind turbines to the grid;
- It reduces the area needed to produce the desired amount of energy, since a large number of turbines is placed in a relatively small area.

Wind farms also have one major disadvantage compared to individually placed turbines: due to the wake behind turbines, an interconnection between turbines is established. These wakes

**Figure 1-1:** A depiction of wind turbine wakes in a wind farm. Source: Christian Steiness.

are shown in Figure 1-1 and have a negative effect on the total power production of a wind farm as well as on the structural loads on the turbines.

Turbines that are located in the wake of an upstream turbine experience, among others, a wind velocity deficit and higher turbulence intensity. This results in lower power production and higher dynamical loads, which can cause fatigue damage. In current wind farms, each turbine is operated at a local optimum. This is referred to as "greedy control". Due to the interaction between turbines, greedy control might be suboptimal, and therefore a more cooperative wind farm control strategy, that takes into account this interaction, is desired. This is why wind farm control has been a popular research topic (for a detailed overview of literature, see, for example, [4], [5] ). The general goal of wind farm control is to optimize not the performance of individual wind turbines, but the wind farm as a whole.

## 1-2 Overview of Wind Farm Modeling and Control

In this section, a brief overview of the state-of-the-art in wind farm modeling and control will be given. Note that the overview given here is far from complete. A more detailed survey on wind farm control can be found in [4]. An overview of wind farm models can, for example, be found in [6], [7], [8], a tutorial on control-oriented modeling and control can be found in [5].

### Wind Farm Modeling

A lot of different control strategies can be applied to improve the performance of wind farms compared to the conventional greedy control. Some of these strategies, such as a Game Theoretic approach [9] or maximum power-point tracking [10], are model-free. However, these model-free control strategies have some important drawbacks. They have to deal with problems such as slow convergence towards the optimum and are difficult to implement on a true wind farm. Therefore, wind farm controllers are in general based on mathematical models, hence the controller performance depends highly on the quality of the model. Choosing a

suitable model for the desired control objective is therefore very import, and should be done deliberately. As a result, many different models have been developed.

Wind farm models exist in a wide range from fast, low-fidelity engineering models to computationally very expensive Computational Fluid Dynamics (CFD) models. Low-fidelity models such as the Park model [11] describe only the dominant wind flow characteristics and are in general relatively simple. Low-fidelity models are static models that use simplifying assumptions to limit the computation time necessary. As a result, they can provide an estimation of the flow quickly, but are less suited for analysis. High-fidelity models such as the Simulator fOr Wind Farm Applications (SOWFA, [12]), on the other hand, capture as much as possible of the flow dynamics and contain, in general, more sophisticated turbine models. These high-fidelity models are generally more accurate than low-fidelity models, but also require much more computational power (up to weeks using distributed computation). So while high-fidelity models provide more accurate wind flow calculations, they are usually not suitable for online model based control purposes where the input must be calculated within one sample period.

**Wind Farm Control**

Wind farm control typically aims to maximize the power production or minimize the structural loadings on the turbines. This can be achieved by individually controlling the turbines in such a way that the collective objective is optimized. Using the degrees of freedom of a wind turbine, two important control methodologies can be distinguished:

- Axial Induction Control (AIC) [13]. In AIC, upstream wind turbines are derated, *i.e.*, the power production of the upstream turbines is reduced. By derating upstream turbines, the wake effects on downstream turbines will decrease. AIC can be worthwhile if negative performance at the derated turbines can be compensated by improved performance of downstream turbines.
- Wake redirection control [14]. In wake redirection control, the rotors of a turbine are yawed with respect to the wind direction in order to deflect the wake. This way, the wake can be steered away from downstream turbines to improve their performance.

In AIC, the Axial Induction Factor (AIF) is used as control variable to derate a turbine. The AIF is defined as the fractional decrease in wind velocity at the turbine with respect to far away from the turbine. It is therefore a measure of the rate at which a turbine slows down the wind. The AIF can be used to determine the power production of a turbine, as well as the forces of the turbine on the flow, and as such is a useful control parameter in wind farm control. AIC is typically used in literature to improve the power production of a wind farm, and is shown to be able to increase the power capture to up to 6% [4].

Wake redirection control can be used to control the centerline of the wake. Hence the performance of downstream turbines can be increased by changing the yaw angle $\gamma$ of upstream turbiens with respect to the wind direction. Note that by yawing a turbine, the performance of this turbine, as in AIC, typically decreases. This loss in performance should again be compensated for by increased performance of downstream turbines. Wake redirection control shows some very promising results (*e.g.* a power increase of up to 7% [4]), but is a relatively new branch of research. The possibilities of wake redirection control are therefore still actively being investigated.

## 1-3 Problem Statement

Most research on wind farm control focusses on steady-state control, *i.e.*, assuming the atmospheric conditions are constant and ignoring dynamic effects such as wake propagation. Dynamic control, optimizing the control input over time for varying conditions, is a relatively underdeveloped branch of wind farm control. Dynamic wind farm control is the next step towards industrialization, since the atmospheric conditions in a wind farm are always subject to change. Steady-state solutions do not capture the inevitably time-varying conditions within a wind farm, and therefore their potential gain is limited. Furthermore, if it is possible to develop a sufficiently fast dynamic controller, real-time control is possible. In real-time control, the next control input is determined while the previous control input is implemented. Dynamic control does however lead to a time-dependent problem, which is in general more complex than the steady-state control problem.

To perform dynamic control on a wind farm, the framework shown in Figure 1-2 will be used. While this framework is well-known within the control community, it is new in the field of wind farm control. The feedback paradigm as depicted in Figure 1-2 has a reference signal $r_k$ at time instant $k$, *e.g.*, a power reference signal. The states $q_k$ can, for example, be the wind flow velocities, while the output $z_k$ can be the dynamic power production of the turbines, but also the forces or stresses on the turbines. The inputs $w_k$ of the wind farm can be the AIFs as well as the yaw angles, as discussed in the previous section. In this thesis, only AIC will be considered. Note that these inputs change the wake, hence influence not only the performance of the turbine in question, but also the performance of the downstream turbines.

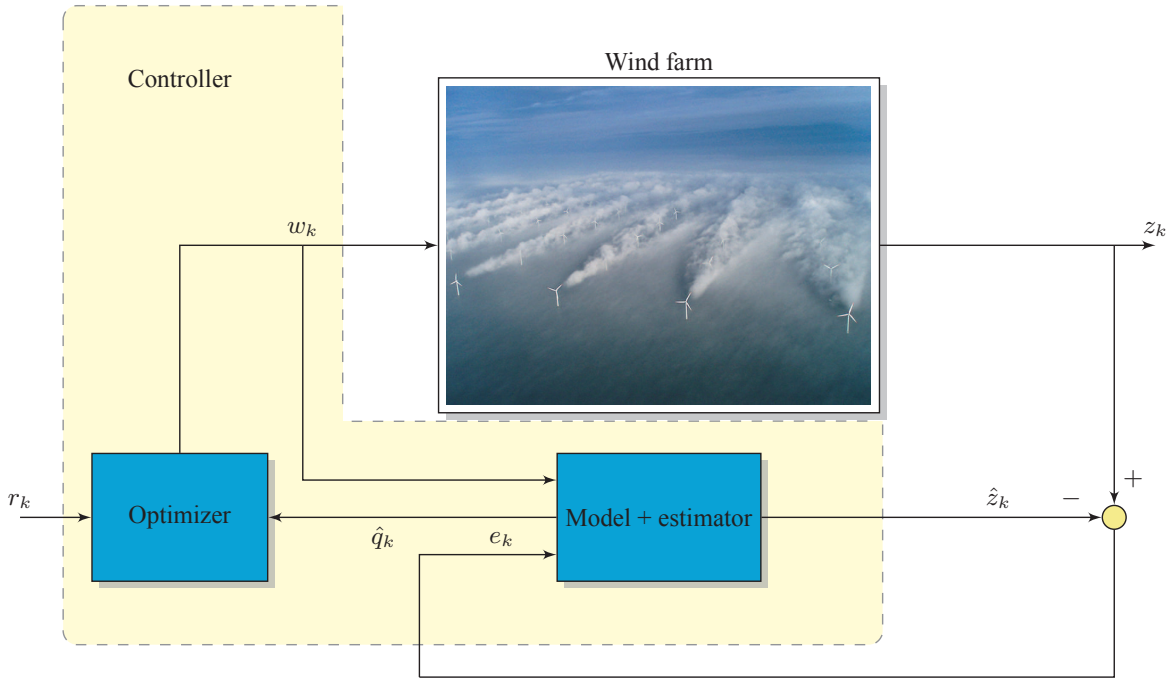



**Figure 1-2:** General closed-loop dynamic control framework.

In Figure 1-2, we can see three important blocks in this framework: the wind farm, the optimizer and the model, possibly complemented with a Kalman estimator [15]. This thesis will focus on the optimizer block in Figure 1-2, in combination with a dynamic control-oriented model: Wind Farm Simulator (WFSim) [16], [17], developed at the Delft University of Technology (DUT). WFSim is a medium-fidelity model based on 2-dimensional (2D) Navier-Stokes (NS) equations. The model simplifies the 3-dimensional flow equations by assuming that, at hub-height, the effect of flow in the z-direction is negligable. By employing a 2D model, the flow in a wind farm can be approximated in an acceptable computation time, enabling dynamic wind farm control.

In WFSim, the 2D NS equations are discretized spatially and temporally on a staggered grid. For each point in this grid, the NS equations are solved to determine the local wind velocities in x- and y-direction, as well as the local pressure. All these velocities and pressures are contained in the state $q_k$ and fed to the controller for optimization. Note that this results in a high-dimensional problem: to guarantee reliable results, the density of the grid should be chosen sufficiently large. This however increases the number of states.

The objective of this thesis is to develop an algorithm according to the framework shown in Figure 1-2, using WFSim as wind farm model. To achieve this, we need to define the objective function and develop an algorithm that can find the optimal control for this objective function. In this thesis, the objective is to maximize the power production of the wind farm over the prediction horizon $N_p$. The resulting problem is an economic Model Predictive Control (MPC) problem [18], which is not yet optimally solved for wind farm scenarios. Note that the prediction horizon needs to be chosen deliberatly: if it is chosen too small, the interaction between turbines through their wakes will not be taken into account. A larger $N_p$ however results in a longer time window that needs to be simulated, leading to longer computation times, as well as a more state vectors $q_k$, $k = 1, 2, \ldots, N_p$. For the grid of 50 by 25 cells typically used in this thesis, the state vector $q_k$ contains 3239 elements *per time step* $k$.

To find the optimum of the objective function, which is unknown and varying over time, an efficient method is needed to determine the gradient of the objective function. A straightforward method to determine the gradient of a function is by using finite differencing. However, since the control input can vary over time, the number of control inputs considered depends on the chosen prediction horizon $N_p$. To determine the gradient with the finite differencing method would require one simulation for each control input, thus leading to a very time-consuming procedure.

An alternative method to determine the gradient is the adjoint method [19], [20]. This method has already been applied on a wind farm in [21], [22], and provides the gradient in a time-efficient manner, enabling dynamic control. With the adjoint method, the gradient can be computed on a standard laptop computer in under a minute. The same calculations would take several hours with the finite differencing method.

Using the gradient, it is possible to determine the optimal control input over time, and implement it over the receding horizon $N_u$. This results in a controller that dynamically maximizes the power production of a wind farm. This leads to the research objective of this thesis:

**With the proposed model-based framework, can we develop a dynamic closed-loop control algorithm that optimizes the power production of a wind farm?**

This research objective can be divided into the following subobjectives:

1. Can we increase the power production of a wind farm with respect to greedy control, using only the AIF of the turbines as input, under constant wind conditions?
2. Can the control algorithm steer the system to an optimal steady-state for constant wind conditions?
3. Can the control algorithm adapt to changing wind conditions?
4. Can the control input be determined fast enough on a standard laptop computer for real-time dynamic control to be possible?

The research objectives given above will be discussed and answered in this report.

## 1-4 Structure of the Report

The structure of the report is as follows: in Chapter 2, the WFSim model will be discussed in more detail. Next, Chapter 3 will discuss the MPC framework that will be used, as well as the challenges this framework provides. Chapter 4 will then present the gradient calculations using the adjoint method. In Chapter 5, the MPC algorithm that has been developed in this thesis will be presented. The simulation results will then be presented in Chapter 6. Finally, Chapter 7 will answer the research objectives given in the previous sections, as well as discuss the future work still to be done in this area.

# Chapter 2

# Wind farm model WFSim

This chapter will be dedicated to the Wind Farm Simulator (WFSim) model that is being developed at Delft University of Technology (DUT) [16],f [17]. WFSim is a two-dimensional (2D) medium-fidelity wind farm model based on the Navier-Stokes (NS) equations that is developed for control purposes.

To decrease the computational expense, WFSim simplifies the three-dimensional (3D) flow equations by assuming that, at hub-height, the effect of flow in the z-direction is negligible. Using this assumption, WFSim solves the NS equations for the flow velocities at hub-height in the wind farm. From these flow velocities, the power production of turbines within the wind farm can be deduced.

There are several reasons why the WFSim model is used in this thesis. First of all, it uses the NS equations to model the flow field, making it a medium-fidelity model. With the simplifying assumptions made, such as the 2-dimensional (2D) simplification, it enables relatively fast flow field computation. This is done by exploitation the sparsity of the matrices, which will be covered in Section 5-1. Furthermore, it is a model that takes into account dynamic effects in the wind flow, as well as changing atmospheric conditions. Finally, it is control-oriented model. Hence it has relatively little tuning parameters and can be implemented effectively for control purposes.

With WFSim, it is possible to implement dynamic control on a wind farm. Most wind farm controllers focus on steady state control [4], even though in an actual wind farm the wind conditions and wake dynamics are likely to change regularly. Due to the relatively fast computations and dynamic flow model of WFSim, it might be possible to design an on-line dynamic controller to optimize the power production of a wind farm.

A schematic representation of WFSim is given in Figure 2-1. In this figure, $q_{atm}$ are the atmospheric conditions, $q$ the states (the velocities and pressures) and $w$ the control inputs of the turbines. $P$ is the power produced by the turbines and $\mathbf{f}$ the forces acting between the turbines and the flow field. Note that the force on the flow is related to the amount of power produced by a turbine.

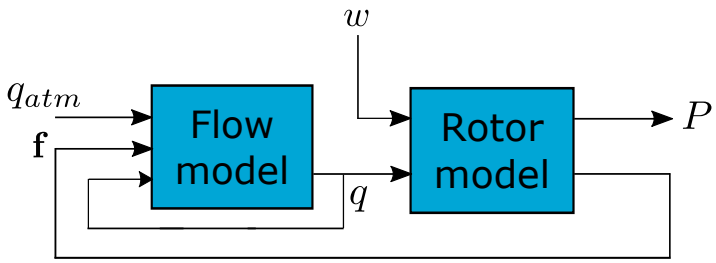



**Figure 2-1:** A schematic representation of the WFSim model.

In the following sections, the WFSim model will be explained. Section 2-1 will cover the wind flow model, and in Section 2-2 the rotor model will be discussed. In Section 2-3, these equations will then be spatially and temporally discretized to give the discrete time wind farm model that will be used for control.

## 2-1 Flow Model

The dynamical behaviour of the flow through a wind farm is described by the NS equations. These equations will be explained in this section. The flow model needs the forces of the turbines exerted on the wind and the inflow wind velocity and direction to compute how the wind flow through the entire field. A schematic representation of the flow model is given in Figure 2-2.

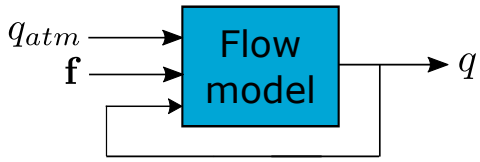



**Figure 2-2:** A schematic representation of the flow model inputs and outputs.

The NS equations comprise of two momentum equations (in x- and in y-direction) and the mass conservation equation. These are given in Equations (2-1) to (2-3).

$$\rho\left(\underbrace{\frac{\partial u}{\partial t}}_{\text{variation}}+\underbrace{\frac{\partial u^2}{\partial x}+\frac{\partial uv}{\partial y}}_{\text{convection}}\right)=-\underbrace{\frac{\partial p}{\partial x}}_{\text{pressure gradient}}+\mu\left(\underbrace{\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}}_{\text{diffusion}}\right)+\underbrace{f_x}_{\text{external forces}} \tag{2-1}$$

$$\rho\left(\underbrace{\frac{\partial v}{\partial t}}_{\text{variation}}+\underbrace{\frac{\partial v^2}{\partial y}+\frac{\partial vu}{\partial x}}_{\text{convection}}\right)=-\underbrace{\frac{\partial p}{\partial y}}_{\text{pressure gradient}}+\mu\left(\underbrace{\frac{\partial^2 v}{\partial y^2}+\frac{\partial^2 v}{\partial x^2}}_{\text{diffusion}}\right)+\underbrace{f_y}_{\text{external forces}} \tag{2-2}$$

$$\rho\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)=0 \tag{2-3}$$

In these equations, $u$ and $v$ [m/s] are the velocities in $x$- and $y$-direction, respectively, and the variable $p$ [Pa] the air pressure. Vectorized, these are the state variables $q = \begin{bmatrix} u & v & p \end{bmatrix}^T$ of the system. The air density $\rho$ is kept constant at 1.2 kg/m$^3$.

For simplicity, the dynamic terms are not included in Equations (2-1) to (2-3). These dynamic terms induce wake recovery, *i.e.*, they ensure that velocity in a wake slowly recovers back to the inflow velocity. To include wake recovery nonetheless, the variable $\mu$ is used as a tuning variable. For normal atmospheric conditions, the dynamic viscosity $\mu << 1$. However, by setting $\mu$ at 10 [kg/ms], wake recovery can be induced, allowing the ommission of a turbulence model in the NS equations.

Please note that dynamic terms are missing in these equations. Instead we use the dynamic viscosity $\mu$, which is a tuning variable, to induce wake recovery. At room temperature, $\mu << 1$. However, by setting $\mu$ at 10 [kg/ms], wake recovery can be induced, which allows us not to include a turbulence model in the NS equations. The external forces $\mathbf{f} = \begin{bmatrix} f_x & f_y \end{bmatrix}^T$ are the forces applied on the flow by the turbines in the farm. These forces will be discussed in Section 2-2.

## 2-2 Rotor Model

In this section, the rotor and turbine model will be discussed. The rotor model determines the thrust forces between rotor and the flow field, which is related to the power produced by this turbine. Figure 2-3 gives the input-output block scheme of the rotor model.

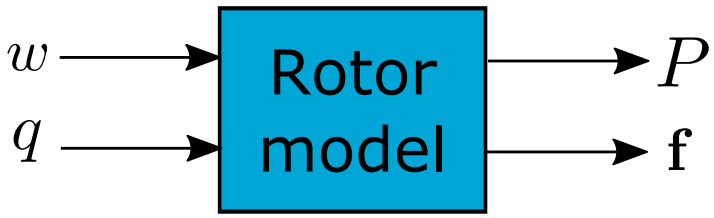



**Figure 2-3:** A schematic representation of the rotor model.

The power produced by a wind turbine is then defined as:

$$P = \frac{1}{2}\rho A_r (U_\infty)^3 C_P(a) \tag{2-4}$$

where $C_P$ is the power coefficient, defined by Betz' law as:

$$C_P = 4a(1-a)^2 \tag{2-5}$$

where $a$ is the Axial Induction Factor (AIF), defined as:

$$a = \frac{U_\infty - U^{r,\perp}}{U_\infty} \tag{2-6}$$

Let us now define a new input, $\beta$, which is the scaled AIF:

$$\beta = \frac{a}{1-a} \tag{2-7}$$

Since it is often impossible to determine $U_\infty$, we want to replace it by the perpendicular velocity at the rotor, $U^{r,\perp}$ in Equation (2-4). $U^{r,\perp}$ is defined as:

$$U^{r,\perp} = \sqrt{u_r^2 + v_r^2}\cos(\gamma - \phi) \tag{2-8}$$

where $\phi$ is the local wind direction with respect to the x-axis and $u_r$ and $v_r$ are the x- and y-component of the wind velocity respectively. If we now replace $U_\infty$ by $U^{r,\perp}$ using Equation (2-6), and $C_P$ with $\beta$ using Equations (2-5) and (2-7), we get an equation that gives the power of a turbine as a function of a few (constant) parameters $\rho$ and $A_r$, as well as the rotor velocity $U^{r,\perp}$ and input $\beta$:

$$P = 2\rho A_r (U^{r,\perp})^3 \beta \tag{2-9}$$

Equation (2-9) shows why we work with $\beta$ as an input instead of $a$: the power $P$ depends linearly on $\beta$. This will be useful when the gradient needs to be calculated. This will be discussed in Chapter 4.

As Figure 2-3 shows, the rotor model also provides the forces of the turbines on the flow. The force $\mathbf{f} = \begin{bmatrix} f_x & f_y \end{bmatrix}^T$ is determined, among others, by the thrust coefficient $C_T$. There are many different types of rotor models known in literature. In WFSim, the rotor of the turbine is modelled using a modified Actuator Disk Model (ADM). Figure 2-4 shows a schematic representation of the wind velocity at the rotor $U^r$ for a given yaw angle $\gamma$ and inflow angle $\phi$. This figure is taken and adapted from [14].

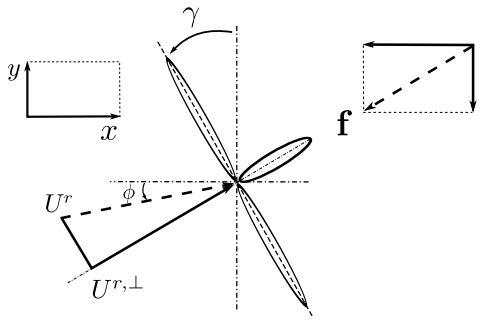



**Figure 2-4:** Schematic representation of the rotor velocity $U^r$ for a given yaw angle $\gamma$ and inflow angle $\phi$, taken and adapted from [14].

In the ADM, the force on a rotor is defined as:

$$f_x = \frac{1}{2}\rho A_r C_T U_\infty^2$$

where $A_r$ is the rotor swept area, which in WFSim equals the rotor diameter since we are dealing with a 2D model. Note that the ADM is originally a 3D model, so this is a 2D

approximation of the ADM. Furthermore, the force term above does not include a yaw angle, but in WFSim, this feature is added nonetheless. Depending on the yaw angle $\gamma$ of a turbine with respect to the wind, the vector $\mathbf{f}$ can be expressed as:

$$\mathbf{f} = \frac{1}{2}\rho A_r C_T U_\infty^2 \begin{bmatrix}\cos(\gamma)\\ \sin(\gamma)\end{bmatrix} \tag{2-10}$$

In Equation (2-2), we again notice the presense of the variable $U_\infty$. Using Equations (2-6) and (2-8), $U_\infty$ can be substituted by $U^{r,\perp}$. This results in the following expression for the force $\mathbf{f}$:

$$\mathbf{f} = \frac{1}{2}\rho A_r C_T(a) \left(\frac{U^{r,\perp}(\gamma)}{1-a}\right)^2 \begin{bmatrix}\cos(\gamma)\\ \sin(\gamma)\end{bmatrix} \tag{2-11}$$

Equation (2-11) gives the force on the flow as a functions of both inputs $a$ (for Axial Induction Control (AIC)) and $\gamma$ (for wake redirection control). As this thesis focusses on AIC, for now the assumption is made that a turbine is always aligned with the wind direction (so $\gamma = 0$). In this case, $f_y = 0$ and $\cos(\gamma) = 1$.

Note that in Equation (2-11), $C_T$ is a function of the AIF $a$. The thrust coefficient $C_T$ plays an important role in determining the forces of turbines on the wind flow. In the ADM, the thrust coefficient is the measure of the force of a wind turbine on the flow as given in equation (2-12).

$$C_T = \frac{f_x}{\frac{1}{2}\rho A_r U_\infty^2} = 4a(1-a) \tag{2-12}$$

where $a$ is the AIF, as defined in Equation 2-6. However, when Equation (2-12) is implemented in WFSim, the power curve, which shows the power production as a function $a$, does not exhibit the desired behavior. From the Betz' law, it is known that the optimal AIF for a single turbine to produce maximal power is $a = \frac{1}{3}$ [23]. Using the definition of $P$ given in Equation (2-4), a power curve as depicted in Figure 2-5 is obtained.

Figure 2-5 shows that the maximal power production is not at $a = \frac{1}{3}$, but rather at $a \approx 0.4$. This is a well-known problem in wind farm modelling when using ADM, and can be resolved by applying the Glauert correction [24]. This correction results in a new definition of the thrust coefficient $C_T$ given in Equation (2-13):

$$C_T(a) = \begin{cases} 4aF(1-a), & \text{if } 0 < a \leq 0.4 \\ \left(8/9 + (4F - 40/9)a + (50/9 - 4F)a^2\right) & \text{if } 0.4 < a < 1, \end{cases} \tag{2-13}$$

where $F$ is a tuning variable, set in this case at 1.38. Using the Glauert correction, the maximal power is now at $a = \frac{1}{3}$, as can be seen in Figure 2-6b.

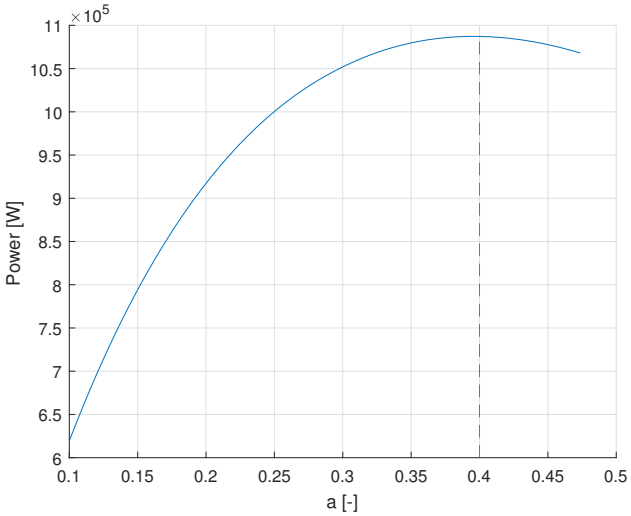
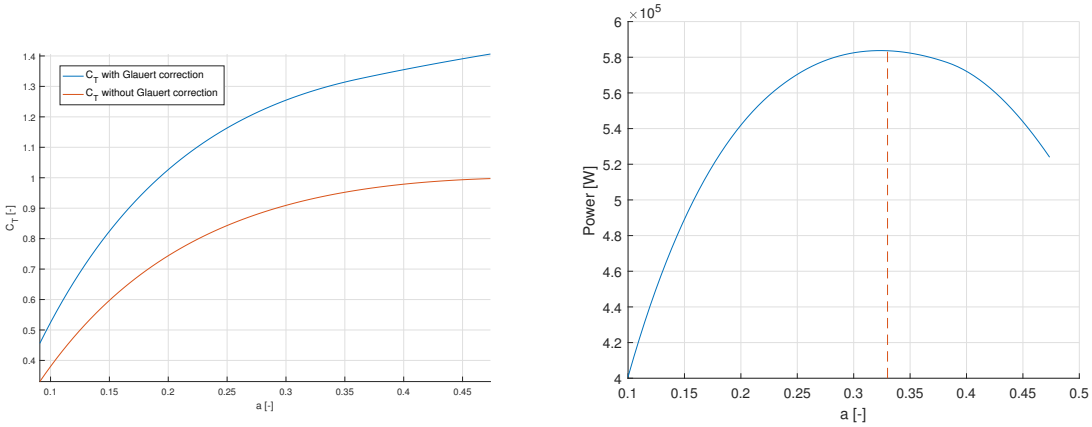



**Figure 2-5:** The power curve in WFSim for a single turbine as a function of $a$.




**(a)** $C_T$ with and without correction.

**(b)** Power curve with correction.

**Figure 2-6:** The $C_T$ curve as function of $a$ with and without the Glauert correction (left) and the power curve with the correction, which shows an optimum of $a = \frac{1}{3}$.

## 2-3 Discretization of WFSim

Now that the flow and rotor model are introduced, the combined system will be discretized. This method will not be fully elaborated here as it was no part of the thesis work, but the basic steps will be shown shortly in this section.

The area of the wind farm will first be divided into a distributed grid of cells. Figure 2-7 shows an example of a cell in such a staggered grid, with the states $u$, $v$ and $p$ shown for this cell.

Next, the NS equations will be discretized for all cells. The spatial discretization is done by using the finite volume method and the temporal discretization by using the implicit method, as explained in [25]. Then, Equations (2-1) to (2-3) are applied on each cell in the grid:

$$\rho \int_{\triangle t} \int_{\triangle V} \left( \frac{\partial u}{\partial t} + (\frac{\partial u^2}{\partial x} + \frac{\partial uv}{\partial y}) \right) dV dt = - \int_{\triangle t} \int_{\triangle V} \left( \frac{\partial p}{\partial x} + \mu \left( \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) + f_x \right) dV dt$$

$$\rho \int_{\triangle t} \int_{\triangle V} \left( \frac{\partial v}{\partial t} + (\frac{\partial v^2}{\partial y} + \frac{\partial vu}{\partial x}) \right) dV dt = - \int_{\triangle t} \int_{\triangle V} \left( \frac{\partial p}{\partial y} + \mu \left( \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right) + f_y \right) dV dt$$

$$\left( \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right) dV dt = 0$$

where $\triangle V$ is the volume of one cell in the grid and $\triangle t$ the sampling time. Note that these equations are nonlinear Partial Differential Equations (PDE's).

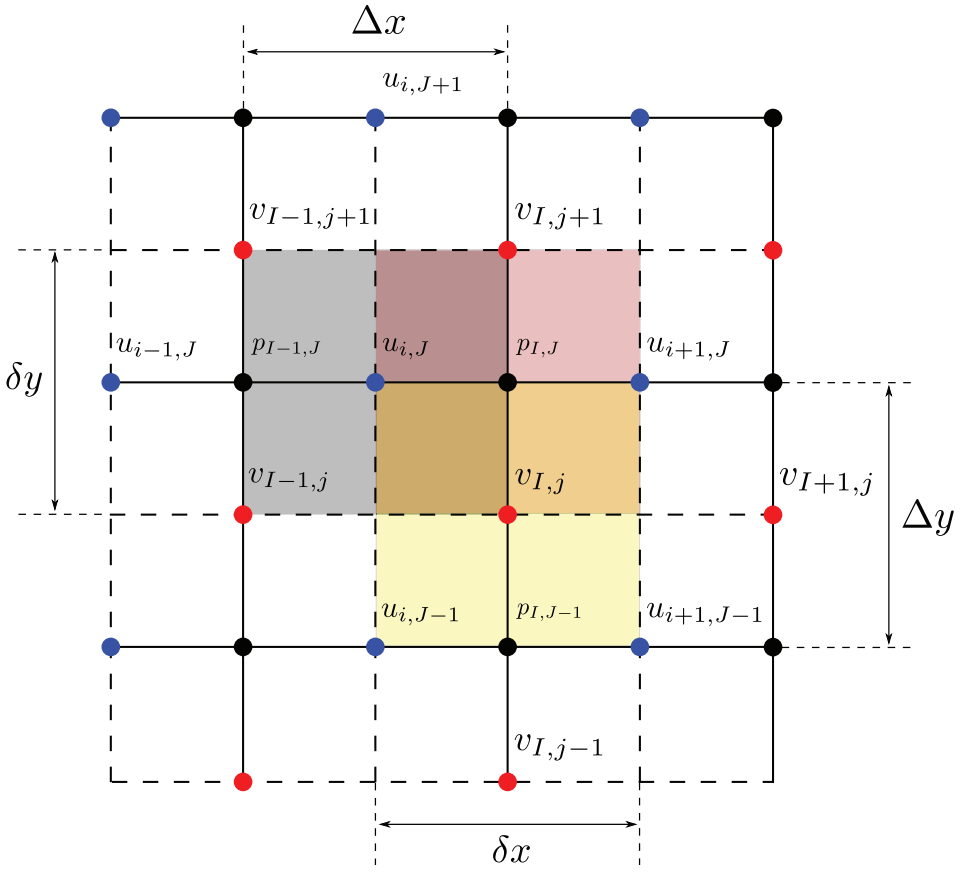



**Figure 2-7:** Example of a cell in WFSim for the x-momentum equation (grey, around $u_{i,J}$), the y-momentum equation (yellow, around $v_{I,j}$ and the continuity equation (pink, around $p_{I,J}$).

Since the grid is linear, each cell in the staggered grid is equivalent. Therefore the volume integral can be taken over the x and y-momentum equations and the continuity equation. This results in the following fully discretized NS equations for each cell:

– x-momentum equation for the $(i,J)^{\text{th}}$ cell (blue in Figure 2-7):

$$a_{i,J}^{px} u_{i,J} = \begin{pmatrix} a_{i,J}^{nx} & a_{i,J}^{sx} & a_{i,J}^{wx} & a_{i,J}^{ex} \end{pmatrix} \begin{pmatrix} u_{i,J+1} \\ u_{i,J-1} \\ u_{i-1,J} \\ u_{i+1,J} \end{pmatrix} - (p_{I,J} - p_{I-1,J})\, \delta y_{j,j+1} + f_{i,J}^{x} \tag{2-14}$$

– y-momentum equation for the $(I,j)^{\text{th}}$ cell (red in Figure 2-7):

$$a_{I,j}^{py} v_{I,j} = \begin{pmatrix} a_{I,j}^{ny} & a_{I,j}^{sy} & a_{I,j}^{wy} & a_{I,j}^{ey} \end{pmatrix} \begin{pmatrix} v_{I,j+1} \\ v_{I,j-1} \\ v_{I-1,j} \\ v_{I+1,j} \end{pmatrix} - (p_{I,J} - p_{I,J-1})\, \delta x_{i,i+1} + f_{I,j}^{y}, \tag{2-15}$$

– continuity equation for the $(I,J)^{\text{th}}$ cell (black in Figure 2-7):

$$0 = \delta y_{j,j+1} \left(u_{i+1,J} - u_{i,J}\right) + \delta x_{i,i+1} \left(v_{I,j+1} - v_{I,j}\right), \tag{2-16}$$

Please note that the states $u_{\bullet,\bullet}$, $v_{\bullet,\bullet}$ and $p_{\bullet,\bullet}$ are defined for time instant $k$, while the coefficients $a_{\bullet,\bullet}^{\bullet}$ and the force terms $f_{\bullet,\bullet}^{\bullet}$ depend on the states at the previous time instant $k-1$. For a detailed definition of these coefficients, see Appendix A.

Next, the states $u_k$, $v_k$ and $p_k$ and the control variables $\beta_k$ and $\gamma_k$ are defined at time step $k$. This results in the following vectors, with $N_x$ the number of grid points in x-direction, $N_y$ the number of grid points in y-direction and $N$ the number of turbines:

$$u_k = \begin{pmatrix} u_{3,2} \\ \vdots \\ u_{3,N_y-1} \\ u_{4,2} \\ \vdots \\ u_{4,N_y-1} \\ \vdots \\ u_{N_x-1,2} \\ \vdots \\ u_{N_x-1,N_y-1} \end{pmatrix}, \quad v_k = \begin{pmatrix} v_{2,3} \\ \vdots \\ v_{2,N_y-1} \\ v_{3,3} \\ \vdots \\ v_{3,N_y-1} \\ \vdots \\ v_{N_x-1,3} \\ \vdots \\ v_{N_x-1,N_y-1} \end{pmatrix}, \quad p_k = \begin{pmatrix} p_{2,2} \\ \vdots \\ p_{2,N_y-1} \\ p_{3,2} \\ \vdots \\ p_{3,N_y-1} \\ \vdots \\ p_{N_x-1,3} \\ \vdots \\ p_{N_x-1,N_y-2} \end{pmatrix}, \quad \beta_k = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_N \end{pmatrix}, \quad \gamma_k = \begin{pmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_N \end{pmatrix}, \tag{2-17}$$

Notice that not all velocities and pressures are present in the state vector. The missing velocities and pressures are the boundary conditions and will be discussed in Section 2-4. Each component in $u_k$, $v_k$ and $p_k$ represents a velocity or pressure at a specific point in the grid defined by the subscript. If we form a grid of the cells from Figure 2-7, a staggered grid as shown in Figure 2-8 is obtained.

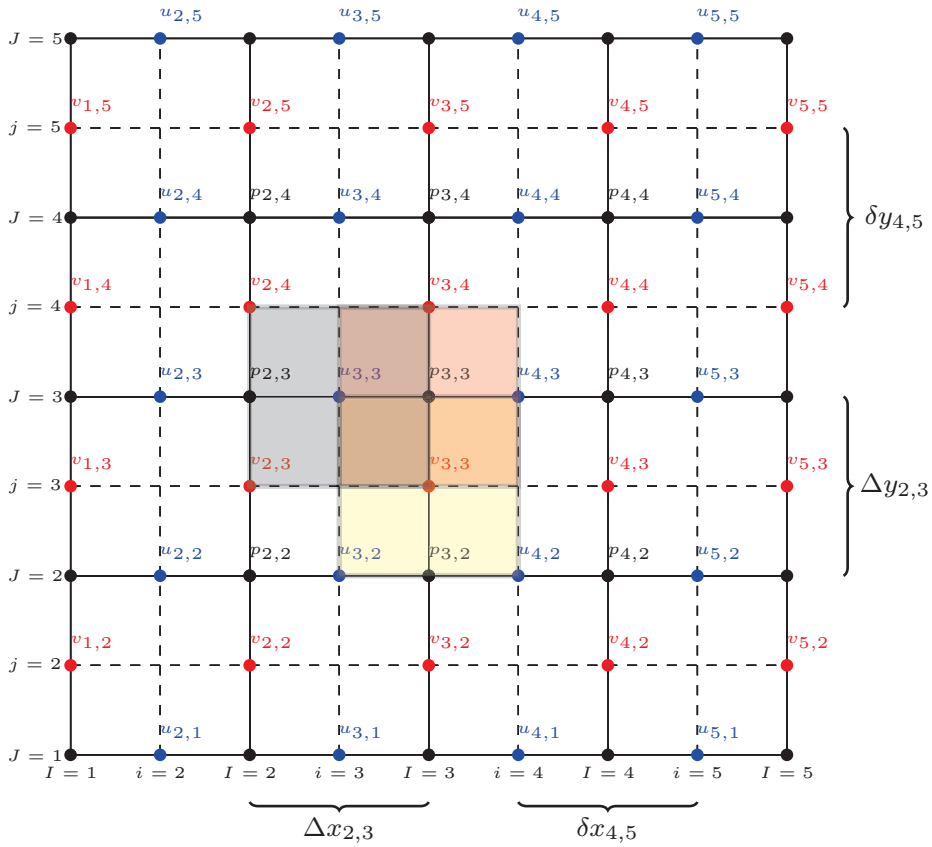



**Figure 2-8:** Example of a staggered grid as used in WFSim, with each cell having volume $\Delta V$.

So finally, the NS equations of Equations (2-1) to (2-3) can be transformed into a discrete nonlinear descriptor model:

$$\underbrace{\begin{bmatrix} A_x(u_{k-1}, v_{k-1}) & 0 & B_1 \\ 0 & A_y(u_{k-1}, v_{k-1}) & B_2 \\ B_1^T & B_2^T & 0 \end{bmatrix}}_{E(q_{k-1})} \underbrace{\begin{bmatrix} u_k \\ v_k \\ p_k \end{bmatrix}}_{q_k} =$$

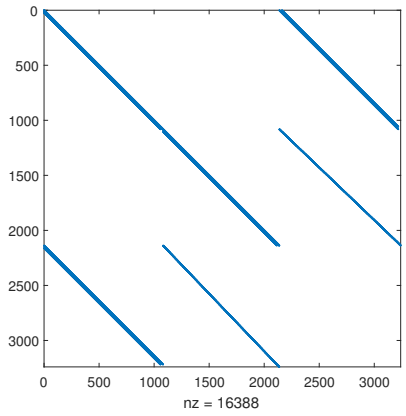
$$\underbrace{\begin{bmatrix} A_{11} & 0 & 0 \\ 0 & A_{22} & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} u_{k-1} \\ v_{k-1} \\ p_{k-1} \end{bmatrix}}_{q_{k-1}} + \underbrace{\begin{bmatrix} b_x(u_{k-1}, v_{k-1}) \\ b_y(u_{k-1}, v_{k-1}) \\ b_c \end{bmatrix}}_{B_c(q_{k-1})} + \underbrace{\begin{bmatrix} f_x(u_{k-1}, v_{k-1}, w_k) \\ f_y(u_{k-1}, v_{k-1}, w_k) \\ 0 \end{bmatrix}}_{S_m(q_{k-1}, w_k)} \tag{2-18}$$

with $B_c$ the vector with the boundary conditions (see Section 2-4) and $S_m$ the vector with the external forces caused by the turbines as discussed in Section 2-2. In Equation (2-18), $E$ is the nonlinear descriptor matrix, which depends on state $q_{k-1}$, while matrix $A$ is constant. Equation (2-18) also is a function of state $q_k$ and input $w_k$:

$$q_k = \begin{bmatrix} u_k \\ v_k \\ p_k \end{bmatrix}, \quad w_k = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_N \end{bmatrix} \tag{2-19}$$

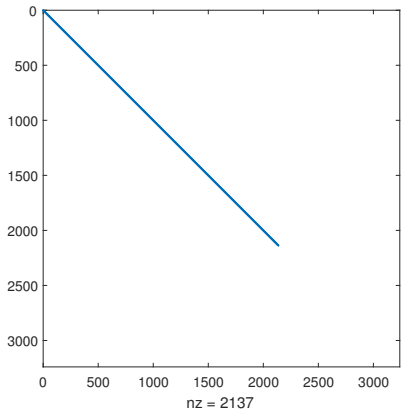
Since in this thesis yaw control will not be covered, the input vector only contains the scaled AIF $\beta$ of all $N$ turbines. From equation (2-18), it should be clear that this system can already become high-dimensional for simple systems. For a relatively small grid of 50 by 25 cells, the system has 3239 states. However, a lot of the elements in Equation (2-18) are zero, making the matrices sparse and therefore the state updates can be calculated relatively fast. This sparsity is shown in Figure 2-9.

[b]




**(a)** Matrix $E$

[b]




**(b)** Matrix $A$

**Figure 2-9:** Nonzero elements in matrices $E$ and $A$ of Equation (2-18)

Figure 2-9 shows all nonzero elements in a typical $E$- and $A$-matrix of Equation (2-18). In this example, the system has 3239 states, resulting in matrices with over $10^7$ elements. In $E$, only 16388 of the $10^7$ elements (0.16%) are nonzero. $A$ has even less nonzero elements, all of which can be found on the diagonal. By exploiting this sparsity, the computation time can be reduced significantly.

## 2-4 Boundary Conditions

As discussed in the previous section, some of the pressure and velocity components in the grid of Figure 2-8 are not included in the state $q_k$. These elements are the elements along the boundaries of the farm, and are therefore determined by the boundary conditions. These are determined by the inflow and outflow. This inflow has a component in x-direction ($u_b$) and a component in the y-direction ($v_b$) and is active on the west side of the grid (see Figure 2-8). Zero stress boundary conditions are prescribed on the other boundaries. This results in the following conditions:

$$
\begin{array}{llll}
u_{2,J} = u_b & \text{for } J = 1, 2, \ldots, N_y, & v_{1,j} = v_b & \text{for } j = 2, 2, \ldots, N_y, \\
u_{i,N_y} = u_{i,N_y-1} & \text{for } i = 3, 4, \ldots, N_x, & v_{I,N_y} = v_{I,N_y-1} & \text{for } I = 2, 3, \ldots, N_x, \\
u_{i,1} = u_{i,2} & \text{for } i = 3, 4, \ldots, N_x, & v_{I,2} = v_{I,3} & \text{for } I = 2, 3, \ldots, N_x, \\
u_{N_x,J} = u_{N_x-1,J} & \text{for } J = 2, 3, \ldots, N_y - 1, & v_{N_x,j} = v_{N_x-1,j} & \text{for } j = 3, 4, \ldots, N_y - 1,
\end{array}
$$

where $N_x$ and $N_y$ are the number of grid points in x and y-direction respectively. The initial conditions are a uniform flow field, *i.e.*, all velocity components $u$ and $v$ are equal to the ambient flow $u_b$, $v_b$. The initial pressure field is set to zero.

Now that the boundary conditions are defined, the system given in Equation (2-18) is now fully defined and can be solved.

## 2-5 The WFSim Algorithm

In the previous sections, the mathematical derivations of the WFSim model have been explained. In this section, a pseudo code is presented to illustrate how the solver computes flow field $q_k$ given $q_{k-1}$ and $w_k$.

First, the user is able to determine the topology of the wind farm by defining the size ($L_x$, $L_y$), the number of grid points ($N_x$, $N_y$), the ambient flow ($u_b$, $v_b$) and the positions of the turbines. If this is done, Equation (2-18) is fully defined and can be solved.

To run the WFSim algorithm, a few other parameters need to be defined. First of all, the step size $\Delta t$ should be defined, the number of time steps $k = N_s$, the convergence parameter $\epsilon$ and the maximum number of iterations $i_{\max}$. When the WFSim algorithm is initiated, it will enter a while-loop until the flow field converges or the maximum number of iterations is reached. Note that choosing $i_{\max} = 1$ results in bypassing the convergence of the wind flow. Then, the algorithm will calculate the propagation of the flow field through time until $k = N_s \cdot \Delta t$.

**Algorithm 1:** Pseudocode for WFSim

$i \leftarrow 1$;
$q_0 \leftarrow$ `Meshing` $(q_{atm}, N_x, N_y, L_x, L_y)$; % Initial flow given the chosen grid
**for** $k = 1 : N_s$ % *Time for-loop until final time* $N_s \cdot \Delta t$
  **while** $\delta > \varepsilon$ & $i < i_{\max}$
    $[E(q_{k-1}), A, B_c(q_{k-1}), S_m(x_{q-1}, w_k)] \leftarrow$ `UpdateSetofEquations` $(q_{k-1}, w_k, \Delta t)$
    % See Equation (2-18)
    $q_k \leftarrow$ `FindSolution` $(E(q_{k-1}), A, B_c(q_{k-1}), S_m(q_{k-1}, w_k))$
    $\delta \leftarrow ||q_k - q_{k-1}||_2$
    $i \leftarrow i + 1$
  **end**
  $i_{\max} \leftarrow 1$
**end**

## 2-6 Linearization of WFSim

Using the system defined in Equation (2-18), it is possible to linearize the system using standard linearization methods. This linear model could then be used to do (local) stability and performance analysis, as well as control design. Although the linear model will not be used for control purposes in this thesis, the derivatives used to form this linearized model are very useful. These partial derivatives are used in the adjoint method to determine the gradient of the cost function. The adjoint method will be discussed thoroughly in Chapter 4.

Equation (2-18) will be linearized around a certain linearization point $\left(q_k^0, w_k^0\right)$. This results in the following linearization:

$$\underbrace{E(q_{k-1}^0)}_{\bar{E}}\delta q_k = \underbrace{\left(A + \left.\frac{\partial B_c(q_{k-1})}{\partial q_{k-1}}\right|_{q_{k-1}^0} + \left.\frac{\partial S_m(q_{k-1}, w_k)}{\partial x_{k-1}}\right|_{x_{k-1}^0, w_k^0} - \frac{\partial E(q_{k-1})}{\partial q_{k-1}}q_k\right)}_{\bar{A}(q_k, q_{k-1}, w_k)}\delta q_{k-1} + \underbrace{\left.\frac{\partial S_m(q_{k-1}, w_k)}{\partial w_k}\right|_{q_{k-1}^0, w_k^0}}_{\bar{B}(q_{k-1}, w_k)}\delta w_k \quad (2\text{-}20)$$

where $\delta q_k = \begin{bmatrix}\delta u_k^T & \delta v_k^T & \delta p_k^T\end{bmatrix}^T$ is the change of state $q_k$, with $\delta u_k = u_k - u_k^0$, $\delta v_k = v_k - v_k^0$ and $\delta p_k = p_k - p_k^0$.

## 2-7 Power Expression in WFSim

In Section 2-2, the power production of a turbine in WFSim has been discussed shortly. The power expression that was given there holds if the turbine coincides with exactly one cell. In this section, the power will be expressed as a function of state $q_k$, as defined in Section 2-3, independent of the size of the turbine. Equation 2-9 in this section gave the power definition $P_{n,k}$ of a turbine as a function of $\beta_{n,k}$ and rotor flow velocity $U_n^{r,\perp}$:

$$P_{n,k} = 2\rho A_r (U_n^{r,\perp})^3 \beta_{n,k} \quad (2\text{-}21)$$

To express this power as a function of state $q_k$, we have to define $U^{r,\perp}$ as a function of $q_k$. Typically, the rotor of a turbine in WFSim will pass through more than one cell. Figure 2-10 shows an example of a small grid with a turbine embedded in it.

In the example of Figure 2-10, the rotor passes through cells $(3,2)$, $(3,3)$ and $(3,4)$. As a result, there is not one state $q_k$ that can equals $U^{r,\perp}$. Instead, we define $U^{r,\perp}$ as the average perpendicular velocity of concerning cells. Assuming no yaw is applied (so $\gamma = 0$), Equation (2-8) becomes:

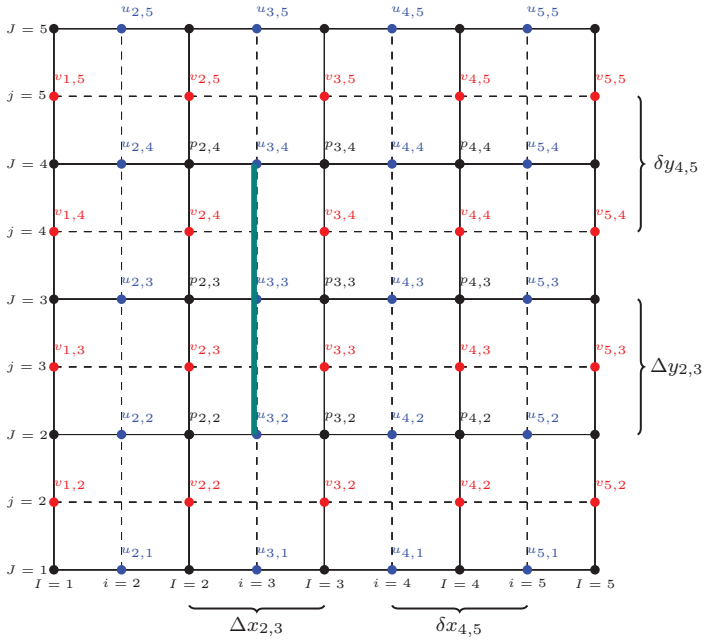



**Figure 2-10:** A small WFSim grid with a turbine embedded in it. The wind flows from west to east here

$$U_n^{r,\perp} = \frac{\sum_{r=1}^{m} \sqrt{u_{n,r}^2 + v_{n,r}^2}}{m} \tag{2-22}$$

where $m$ is the number of cells coinciding with the turbine. So in the example of Figure 2-10, $m = 3$, $u_{n,r=1} = u_{i=3,J=2}$, $u_{n,r=2} = u_{i=3,J=3}$ and $u_{n,r=3} = u_{i=3,J=4}$. These velocities are, as we can see using the definition of the state $q_k$ defined in Equations (2-19), part of the state vector. So now it is also possible to write the power of turbine $n$ as a function of $q_k$ and $w_k$:

$$P_{n,k} = 2\rho A_r \left( \frac{\sum_{r=1} m\sqrt{u_{n,r}^2 + v_{n,r}^2}}{m} \right)^3 \beta_{n,k} \tag{2-23}$$

Notice that only a very limited amount of states are actually in the expression of the power $P_{n,k}$. For the example of Figure 2-10, only 3 of the 25 cells coincide with the rotor. Usually, a wind farm is significantly larger than this example, so an even smaller ratio of states would appear in the expression for $P_{n,k}$.

To summarize, we have seen in this chapter how the 2D NS equations are used to obtain a wind farm model. This model is discretized spatially and temporally and the expressions for the forces on the turbines and the power produced by the turbines are given. In the following chapters, this model will be used to perform Model Predictive Control (MPC).

# Chapter 3

# Model Predictive Control for Wind Farm Applications

In this chapter, we will discuss Model Predictive Control (MPC). If we look back at Figure 1-2, we see two important blocks in the dynamic control framework: the wind farm model (plus estimator) and the optimizer. The model is discussed thoroughly in Chapter 2. Since we do not have access to a wind farm, the model discussed here will also be considered as the wind farm under study. An estimator is ommitted in this thesis. Instead we assume full state knowledge.

This leaves the optimizer. In this thesis, nonlinear Model Predictive Control will be used to develop a control that can dynamically optimize the power production of a wind farm using the Wind Farm Simulator (WFSim) model. Section 3-1 will give a short introduction into economic MPC, followed by an overview of the MPC framework that will be used in this thesis.

## 3-1 Economic MPC

In MPC a cost function $J$ is defined, and the MPC algorithm will then optimize the control inputs $w$ over a given finite control horizon. For now, it is assumed that the control horizon $N_c$ is equal to prediction horizon $N_p$. The algorithm will thus search for the input $w$ that minimizes this cost function. In classic MPC, the (quadratic) cost function is usually defined as:

$$J = \sum_{k=1}^{N_p} (q_k - q_k^*)^T Q (q_k - q_k^*) + \sum_{k=1}^{N_p} (w_k)^T R (w_k) \tag{3-1}$$

This cost function has two terms: one involving the state $q$, and the other penalizing input $w$. So in this case, the cost function $J$ would be minimal if $q_k = q^*$ and $w_k = 0$. $Q \in \mathbb{R}^{n_q \times n_q}$

and $R \in \mathbb{R}^{n_w \times n_w}$ are then weighing matrices that determine the importance of certain states $q$ and inputs $w$ respectively. The assumption here is that there is some reference signal or optimal steady-state $q^*$ that is known a-priori. The MPC algorithm will then steer the system to the steady-state that minimizes Equation (3-1).

As discussed in Chapter 1, the goal of this thesis is to maximize the power production of wind farms dynamically. This means that the optimizer should be able to maximize the power production over time for varying wind conditions. It is virtually impossible to determine the optimal power production for a wind farm under *all possible* wind conditions. Hence we assume no a-priori knowledge of the maximal power production for given atmospheric conditions. This implies that it is not possible to define the cost function as defined in equation (3-1). Instead, the cost function will look as shown in Equation (3-2).

$$J = -\sum_{k=1}^{N_p} P_k \tag{3-2}$$

with $P_k$ the power production of the wind farm at time instant $k$. Note that Equation (3-2) does not include a penalty on the input. This term is ommitted since the main focus in this thesis is to maximize the power production. If a penalty on the input were included, this could result in a lower power production in favor of the control signal. It can however be included when desired.

The cost function defined in Equation (3-2) is not quadratric and subsequently the optimum of $J$ does not go towards 0, *i.e.*, Equation (3-2) may not take its optimal value at steady state. This makes the problem considered in this thesis fundamentally different from standard MPC problems, see also [18].

These types of problems belong to the class of *Economic MPC*, a relatively new research area. Many papers can be found that cover economic MPC, usually for chemical plants [26], [18], but also for power management [27]. However, in most of them, a steady-state terminal constraint is known [18], which in this application is not the case. It is therefore a scientific field in which further research needs to be done.

One of the problems that occurs with economic MPC will be discussed more extensively in Section 4-5. To find the optimum of the objective function, the gradient of $J$ can be used. This gradient gives the direction in which the input has to move to decrease $J$. More on the gradient can be found in Chapter 4.

Figure 3-1 shows a typical gradient for a 3-turbine wind farm over one prediction horizon $N_p = 400$ s for a 3-turbine wind farm in WFSim. It clearly exhibits a steep drop at the end of the prediction horizon, meaning the MPC algorithm steers the inputs to the maximum allowed value. This has two causes, both having more or less the same underlying reason: the objective function tries to maximize the power production within the prediction horizon. As a result, it completely ignores the power production just *after* the prediction horizon. These are the causes:

1. Both the power $P_k$ and the state $q_k$ are functions of state $q_{k-1}$;
2. A non-quadratic objective function results in a controller that does not tend to steer towards steady state.

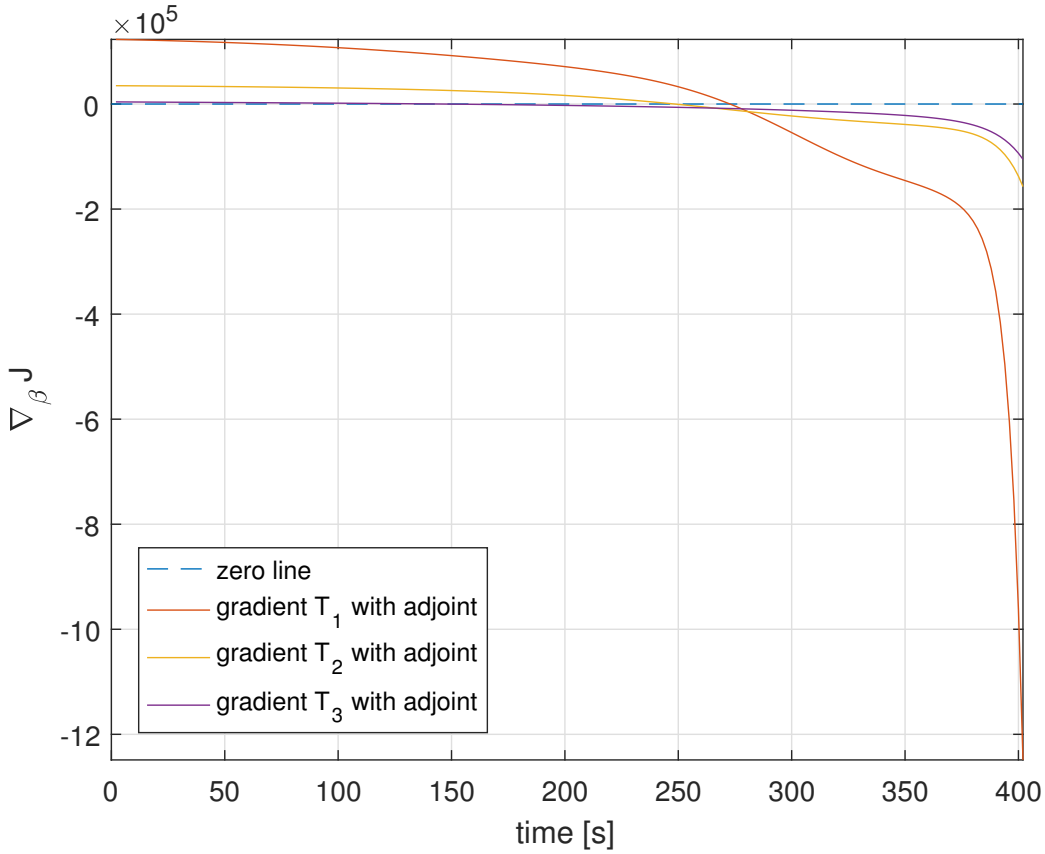



**Figure 3-1:** A typical gradient $\bigtriangledown J$ of Equation (3-2).

As a result of the first cause, there is a lag on the power production. For an $N_p$ of 1, the optimal input would always be to increase the control signal as much as possible, since the power is determined by state $q_0$ and input $w_1$ (see Equation (2-4)). It therefore doesn't matter how much the wind is slowed down at time $q_1$, since this doesn't influence the power production within the horizon. The same thing happens at the end of a larger prediction horizon, as noticable in Figure 3-1.

Since the objective function is non-quadratic, the algorithm wants to increase the power production at the end of the prediction horizon by increasing the input. Therefore, as soon as the wind will not reach downwind turbines anymore within the prediction horizon, there is no more interaction between different turbines. Hence the gradient will steer its input to its local optimum. More on this can be found in Section 4-5.

Possible solutions to this problem are infinite horizon MPC [28] or an end-point constraint [29]. In infinite horizon MPC, a term is added to the objective function that represents the power beyond the prediction horizon. This is difficult to implement since it would mean that the final value would become even more important. This could then amplify the problem of increasing input at the end of the prediction horizon. Furthermore, an end-point constraint is difficult to implement since it is not straightforward what the end-point constraint should be since the optimal power is unknown to us.

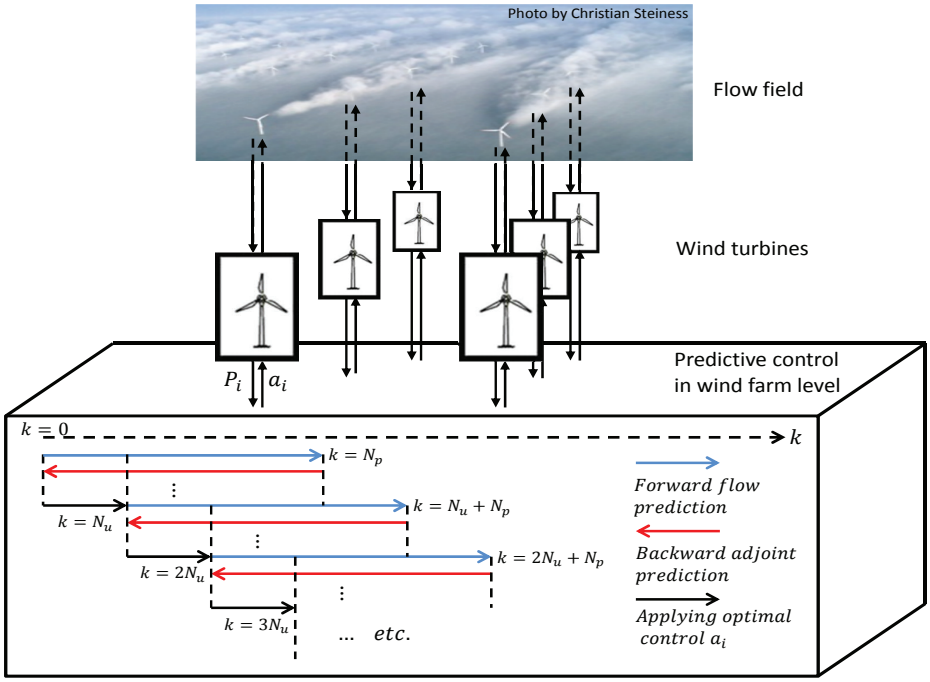



**Figure 3-2:** The MPC framework used in this thesis, taken from [22].

## 3-2 The MPC Framework

In this section, the concept of the MPC framework that is used in this thesis will be presented. It is closely related to the work done in [21] and [22], but does differ in some significant details. This will be discussed in more detail in Chapter 5. The framework used in this thesis is presented in Figure 3-2.

On the top, we see the flow field in the wind farm. As we assume full state knowledge, we have a-priori knowledge of this flow field. This field interacts with the turbines within the field, shown in the middle of Figure 3-2. The turbines get input from the MPC algorithm and give feedback by means of the power production.

The bottom block represents the optimization algorithm that uses MPC. First, the flow of the wind is simulated for the next $N_p$ seconds using an initial input sequence. This also gives a first calculation of the power production over $N_p$, so the cost function $J$ from Equation (3-2) can be calculated. The algorithm then needs to find an input sequence $\tilde{w}$ that decreases $J$, *i.e.*, increases the total power production over the prediction horizon $N_p$. Note once more that the control horizon is set at $N_c = N_p$. This leads to the following objective function:

$$\min_{\tilde{w}} \quad J(\tilde{q}, \tilde{w}) = \min_{\tilde{w}} \quad \sum_{k=1}^{N_p} -P_k(q_{k-1}, w_k) \tag{3-3}$$

Notice that Equation (3-3) contains states and inputs at each time instant $k = 1, 2, \cdots, N_p$. We therefore introduce the concatenated state vector $\tilde{q}$ and input vector $\tilde{w}$:

$$\tilde{q} = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_{N_p} \end{bmatrix}, \qquad \tilde{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_{N_p} \end{bmatrix} \tag{3-4}$$

A commonly used method to determine what input $\tilde{w}_k$ would decrease the function $J$ is by determining the derivative $\frac{dJ}{d\tilde{w}}$. This gradient will give the direction in which $\tilde{w}$ should move to improve $J$. This is very important if we want to optimize the input $\tilde{w}$, since without the gradient it would be unknown how the input would need to change to improve the cost function. Note that the number of inputs for this problem is $N_p \cdot N$, with $N$ the number of turbines, all of which need to be optimized. So for a 3-turbine wind farm with a prediction horizon of 300 time steps, this would already result in a gradient with 900 elements.

This introduces a challenge, as it is not straightforward to calculate this derivative. As we have seen in Chapter 2, the power of a turbine in WFSim depends on the rotor velocity $U^{r,\perp}$ and input $w = \beta$, but $U^{r,\perp}$ also depends on input $\beta$ in a nonlinear fashion. Furthermore, changing input $w_k$ at time instant $k$ might not only change the power $P_k$ at time instant $k$, but also at later time instants.

It is therefore important to find a reliable and time efficient way to determine this derivative $\frac{dJ}{d\tilde{w}}$. The most straightforward way to determine the gradient is by applying finite differencing. In this method, the inputs are excited individually with a small value $\delta$ to determine change of power this induces. Then the derivative can be determined by using Equation (3-5):

$$\frac{dJ}{dw_{k,n}} = \frac{J(w_{k,n} + \delta) - J(w_{k,n})}{\delta} \tag{3-5}$$

where $w_{k,n}$ is the input of turbine $n$ at time instant $k$. To obtain the gradient $\bigtriangledown_{\tilde{w}} J$, $J(w_{k,n}+\delta)$ needs to be determined for each element in $\tilde{w}$. For the example mentioned above, with 3 turbines and $N_p = 300$ time steps, this would mean 900 extra forward simulations are needed to determine the gradient. This would of course take a lot of computation time. This will be further elaborated in Section 4-5.

It is therefore essential to find an alternative method to determine the gradient in a more time-efficient manner. One way to do this is by applying the adjoint method [19], [20]. In [21], [22], the adjoint method is also used for power optimization in a wind farm. This method will be discussed thorougly in Chapter 4.

If the gradient $\bigtriangledown_{\tilde{w}} J = \frac{dJ}{d\tilde{w}}$ is determined, the optimal step size $\alpha$ needs to be found, resulting in a new input $\tilde{w}^*$:

$$\tilde{w}^* = \tilde{w} - \alpha \cdot \bigtriangledown_{\tilde{w}} J \tag{3-6}$$

The optimal value for $\alpha$ can be found by doing a line-search. By doing a forward simulation using WFSim for each step size $\alpha$, and subsequently evaluating the cost function $J$, the optimal step size is determined. Note that it might be necessary to limit the number of line searches to prevent endless simulations. This implies that the optimal step size $\alpha$ might not be found in one line search.

When the step size is optimized, the found input $\tilde{w}^*$ is implemented on the turbines in the wind farm for $N_u$ time steps. Meanwhile, the MPC algorithm will again do a forward simulation and corresponding gradient calculation to optimize the input for the next prediction horizon. This is called the receding horizon principle and is shown in Figure 3-2.

## 3-3 Problem Statement

As mentioned above, the goal is to maximize the power production of a wind farm using WFSim. Converted to a mathematical expression and using Equation (2-4), this gives an objective function $J$ as given in Equation (3-3). The goal of the algorithm is to find a control input that maximizes the power production $P$ of a wind farm.

It is often difficult to determine the gradient of the cost function. A method to determine the full derivative $\frac{dJ}{d\tilde{w}}$ of cost function $J$ with respect to input $\tilde{w}$ is by using the adjoint method. This method exploits the fact that the relationship between $\tilde{w}$ and $\tilde{q}$ is known. In our case, it is given in Equation (2-18). We can rewrite this equation into an equality constraint $C_k(q_k, w_k) = 0$:

$$C_k(q_k, q_{k-1}, w_k) = E(q_{k-1}, q_k) - Aq_{k-1} - B_c(q_{k-1}) - S_m(q_{k-1}, w_k) \ \in \ \mathbb{R}^{n_q \times 1} \tag{3-7}$$

where $n_q$ is the number of states. Equation (3-7) has to be fulfilled over the entire prediction horizon $N_p$. This yields the following concatenated constraint matrix:

$$C(\tilde{q}, \tilde{w}) = \begin{bmatrix} C_1(q_0, w_o, q_1, w_1) \\ C_2(q_1, w_1, q_2, w_2) \\ \vdots \\ C_{N_p}(q_{N_p-1}, w_{N_p-1}, q_{N_p}, w_{N_p}) \end{bmatrix} \in \mathbb{R}^{N_p \cdot n_q \times 1} \tag{3-8}$$

resulting in the following Partial Differential Equation (PDE) constraint optimization problem:

$$\begin{aligned} \min_{\tilde{w}} \quad & J(\tilde{q}, \tilde{w}) \quad = \min_{\tilde{w}} \quad \sum_{k=1}^{N_p} -P_k(q_{k-1}, w_k) \\ s.t. \quad & C(\tilde{q}, \tilde{w}) \qquad = 0 \end{aligned} \tag{3-9}$$

The adjoint method uses the partial derivatives of both $J$ and $C$ in Equation (3-9) with respect to $\tilde{q}$ and $\tilde{w}$ to compute the gradient $\frac{dJ}{d\tilde{w}}$. The following chapter will cover the adjoint method and how it is used to determine the gradient of Equation (3-2). Subsequently, Chapter 5 will cover the algorithm based on the framework discussed in this chapter.

# Chapter 4

# Gradient Calculation using the Adjoint Method

As explained in Chapter 3, the goal of this thesis is to develop an algorithm that can optimize the power production of a wind farm using Wind Farm Simulator (WFSim). As discussed there, a method needs to be found to determine the gradient of the cost function $J$. In this chapter, detailed explanations will be given of the underlying mathematics that concern the adjoint method, and how it is used to determine the gradient of our cost function $J$.

## 4-1 General Derivation of the Adjoint Method

In this section, the general derivation of the adjoint method for finding the gradient of function $J$ is explained. For clarity, the time index dependency is dropped here. The formulation given here holds for both continuous and discrete time.

For now, we assume that we have some cost function $J$ that is a function of both state $q$ and input $w$. Furthermore, we have a constraint equation $C(q, w) = 0$ defined by Partial Differential Equations (PDEs) that describe the system we want to control. We then have the following PDE-constrained optimization problem:

$$\begin{aligned} \min_{w} \quad & J(q, w) \\ \text{s.t.} \quad & C(q, w) = 0 \end{aligned} \tag{4-1}$$

Note that the problem defined above is a more general form of Equation (3-9). To find the optimum of this cost function, the gradient of $J$ with respect to control input $w$ can be used. We can then use the constraint equation $C$ to simplify this. The total derivative of the constraint will also be zero for any $(q, w)$, as $C$ is zero for all $q$ and $w$. So:

$$\frac{dC}{dw} = \frac{\partial C}{\partial w}\frac{dw}{dw} + \frac{\partial C}{\partial q}\frac{dq}{dw} = \partial_w C + \partial_q C \partial_w q = 0 \tag{4-2}$$

where $\partial_\bullet$ is short for the partial derivative $\frac{\partial}{\partial \bullet}$. The total derivative of the cost function $J(q, w)$ is:

$$\frac{dJ}{dw} = \bigtriangledown_{\tilde{w}} J = \frac{\partial J}{\partial w}\frac{dw}{dw} + \frac{\partial J}{\partial q}\frac{dq}{dw} \tag{4-3}$$

Looking at Equation (4-3), we notice that not all terms are straightforward to calculate. More specifically, the derivative $\frac{dq}{dw}$ is very hard to determine, since the relationship between $q$ and $w$ is described by a PDE. This is where the adjoint method can help generate a solution.

Since our constraint equation should always equal zero, we can merge it into our cost function without this affecting its outcome. To do this, we use the Lagrange multiplier $\lambda$ to get the Lagrangian $\mathscr{L}$:

$$\mathscr{L}(q, w, \lambda) = J(q, w) + \lambda^T C(q, w) \tag{4-4}$$

Since constraint $C$ is always zero, the Langrangian equals our original cost function $J$. As a result, we can say that the derivative of $J$ is also equal to the derivative of $\mathscr{L}$, with the added benefit that we can choose $\lambda$ freely since $C = 0$. The derivative of the Lagrangian then becomes:

$$\begin{aligned}\frac{dJ}{dw} &= \frac{d\mathscr{L}}{dw} \\ &= \frac{\partial J}{\partial w} + \frac{\partial J}{\partial q}\frac{dq}{dw} + \underbrace{\frac{d}{dw}(\lambda^T)C}_{=0} + \lambda^T(\frac{\partial C}{\partial w} + \frac{\partial C}{\partial q}\frac{dq}{dw})\end{aligned}$$

We can see immediately that the third term drops out since $C = 0$. This gives us, after rearranging, the following equation:

$$\begin{aligned}\frac{dJ}{dw} &= \frac{\partial J}{\partial w} + \left(\frac{\partial J}{\partial q} + \lambda^T \frac{\partial C}{\partial q}\right)\frac{dq}{dw} + \lambda^T \frac{\partial C}{\partial w} \\ &= \partial_w J + \left(\partial_q J + \lambda^T \partial_q C\right)\partial_w q + \lambda^T \partial_w C\end{aligned} \tag{4-5}$$

Then, we can exploit the freedom in $\lambda$ by defining the following adjoint equation:

$$\left(\partial_q C\right)^T \lambda = -\left(\partial_q J\right)^T \tag{4-6}$$

If we use Equation (4-6) on Equation (4-5), the terms between the brackets become zero, resulting in the following total derivative of $J$:

$$\frac{dJ}{dw} = \partial_w J + \lambda^T \partial_w C \tag{4-7}$$

So now, it is possible to calculate the gradient of $J$ with respect to input $w$ by using the partial derivatives of $J$ and $C$ with respect to both input $w$ and state $q$ respectively. In

comparison to Equation (4-3), it is no longer necessary to calculate the $\frac{dq}{dw}$-term, which in general is difficult to obtain. The following sections will show how this method can be used to determine the gradient of $J$ for a simple single turbine example, as well as for a high-dimensional time-dependent wind farm problem.

## 4-2 Illustrative Single Turbine Example

To show that the adjoint method produces a correct derivative, we will present an illustrative single wind turbine example. The objective is to maximize the total power production of this turbine, so, using Equation (2-9) the objective function becomes:

$$J(\beta, U^{r,\perp}) = -2\rho A_r (U^{r,\perp})^3 \beta \tag{4-8}$$

If we now want to find the total derivative of Equation (4-8) with respect to the control input $\beta$, we can do this in three different ways:

1. Express $U^{r,\perp}$ as a function of $\beta$, and substitute this into Equation (4-8). We now have a function that only depends on $\beta$, so we can use the direct derivative $\frac{dJ}{d\beta}$ to determine the optimum;
2. Take the total derivative of Equation (4-8) to find the optimum;
3. Use the adjoint method to find the optimum.

This section will show that all three methods yield the same result.

**Gradient using substitution of non-input-related terms**

We can express equation (4-8) as a function of only $\beta$ by substituting $U^r = \frac{U_\infty}{1+\beta}$. This yields:

$$J(\beta) = -2\rho A_r U_\infty^3 \frac{\beta}{(1+\beta)^3}$$

Now we can determine the derivative:

$$\begin{aligned} \frac{dJ}{d\beta} &= -2\rho A_r U_\infty^3 \frac{1-2\beta}{(1+\beta)^4} \\ &= -2\rho A_r (U^r)^3 \frac{1-2\beta}{1+\beta} \end{aligned} \tag{4-9}$$

**Gradient using the total derivative**

If we use the total derivative, as shown in Equation (4-10), we should get the same result as in Equation (4-9). The total derivative for a function with 2 variables is, by definition, given as:

$$\frac{df(x,t)}{dt} = \frac{\partial f(x,t)}{\partial x}\frac{dx}{dt} + \frac{\partial f(x,t)}{\partial t} \tag{4-10}$$

If we now apply Equation (4-10) on the cost function, we get:

$$\begin{aligned}\frac{dJ(U^r,\beta)}{d\beta} &= \frac{\partial J(U^r,\beta)}{\partial U^r}\frac{dU^r}{d\beta} + \frac{\partial J(U^r,\beta)}{\partial \beta} \\ &= \left(-6\rho A_r (U^r)^2\beta\right)\cdot\left(-\frac{U_\infty}{(1+\beta)^2}\right) - 2\rho A_r (U^r)^3 \\ &= 6\rho A_r (U^r)^3\frac{\beta}{1+\beta} - 2\rho A_r (U^r)^3 \\ &= -2\rho A_r (U^r)^3\frac{1-2\beta}{1+\beta}\end{aligned}$$

Which corresponds to the answer found in Equation (4-9).

**Gradient using the adjoint method**

Finally, we are going to use the adjoint method to determine the derivative of $J$. To use this method, we need to write the expression of $U^r$ as an equality constraint.

$$C(U^r,\beta) = U^r - \frac{U_\infty}{1+\beta} = 0 \tag{4-11}$$

If we now calculate the partial derivatives of Equations (4-1) and (4-11) with respect to state $U^r$, we get:

$$\partial_{U^r} J = -6\rho A_r (U^r)^2\beta$$

and

$$\partial_{U^r} C = 1$$

which results, using Equation (4-6), in a $\lambda$ of:

$$\lambda = -C_{U^r}^{-T} J_{U^r} = 6\rho A (U^r)^2\beta$$

Next, we need the derivatives of $C$ and $J$ with respect to input $\beta$:

$$\partial_\beta J = -2\rho A (U^r)^3$$

and

$$\partial_\beta C = \frac{U_\infty}{(1+\beta)^2}$$

which, using Equations (2-8) and (4-7), leads to the following total derivative:

$$\begin{aligned}\frac{dJ(U^r,\beta)}{d\beta} &= \lambda^T C_\beta + J_\beta \\ &= (6\rho A(U^r)^2\beta)\cdot\left(\frac{U_\infty}{(1+\beta)^2}\right) - 2\rho A(U^r)^3 \\ &= -2\rho A(U^r)^3\frac{1-2\beta}{1+\beta}\end{aligned}$$

which again is the same expression as found in the previous subsection, thus showing that the adjoint method indeed gives the same derivative of the cost function for this example.

## 4-3 Time-dependent Problems

In the example of the previous section, the objective function was not time-dependent. As explained in Chapter 1, the goal of this thesis is to design a dynamic controller that tries to find the optimal control over a receding horizon using a Model Predictive Control (MPC) approach. This results in a time-dependent problem where the goal is to optimize the power production over the entire prediction horizon $N_p$.

Since we now simulate over time, we also get different states $q$ and inputs $w$ at different time instants $k$. We define $q_k$ as the state at time instant $k$, and similarly $w_k$ is the input at time instant $k$. This results in a concatenated state and input vector as shown in Equation (3-4). The optimization problem is then defined in Equation (3-9).

We have seen in Section 4-1 that we need the partial derivatives of $C$ to determine the gradient $\bigtriangledown_{\tilde{w}} J$. Looking at Equation 3-8, we see that $\partial_{\tilde{w}} C$ and $\partial_{\tilde{q}} C$ become more complicated. These partial derivatives now also become stacked matrices:

$$\partial_{\tilde{q}} C = \begin{bmatrix} (C_1)_{q_1} & 0 & & \\ (C_2)_{q_1} & (C_2)_{q_2} & 0 & \\ 0 & \ddots & \ddots & \ddots \\ & 0 & (C_{N_p})_{q_{N_p-1}} & (C_{N_p})_{q_{N_p}} \end{bmatrix} \in \mathbb{R}^{N_p\cdot n_q \times N_p \cdot n_q} \tag{4-12}$$

where $(C_1)_{q_1}$ is the partial derivative of $C_1$, the constraint at time instant $k=1$, with respect to state $q_1$, etc. Notice that this can become a very large matrix: $C_k$ has size $(n_q \times 1)$, with $n_q$ the size of vector $q$ at time instant $k$.

Similarly:

$$\partial_{\tilde{w}} C = \begin{bmatrix} (C_1)_{w_1} & & & \\ (C_2)_{w_1} & (C_2)_{w_2} & & \\ & \ddots & \ddots & \\ & & (C_{N_p})_{w_{N_p-1}} & (C_{N_p})_{w_{N_p}} \end{bmatrix} \in \mathbb{R}^{N_p\cdot n_q \times N_p \cdot n_w} \tag{4-13}$$

with $n_w$ the size of input vector $w$ at any given time instant $k$. The partial derivative of $J$ is then defined as:

$$\frac{dJ}{d\tilde{q}} = \begin{bmatrix}(J)_{q_1} & (J)_{q_2} & \cdots & (J)_{q_{N_p}}\end{bmatrix} \in \mathbb{R}^{1\times N_p \cdot n_q} \tag{4-14}$$

$$\frac{dJ}{d\tilde{w}} = \begin{bmatrix}(J)_{w_1} & (J)_{w_2} & \cdots & (J)_{w_{N_p}}\end{bmatrix} \in \mathbb{R}^{1\times N_p \cdot n_w} \tag{4-15}$$

Subsequently, also the Lagranian $\Lambda$ will be a stacked matrix: $\Lambda = \begin{bmatrix}\lambda_1 & \lambda_2 & \dots & \lambda_{N_p}\end{bmatrix}^T \in \mathbb{R}^{N_p \cdot n_q \times 1}$, creating the adjoint equation $(\partial_q C)^T \Lambda = (\partial_q J)^T$, which comes down to:

$$\begin{bmatrix} (C_1)_{q_1}^T & (C_2)_{q_1}^T & & & & \\ & (C_2)_{q_2}^T & (C_3)_{q_2}^T & & & \\ & & \ddots & \ddots & & \\ & & & (C_{N_p-1})_{q_{N_1}}^T & (C_N)_{q_{N_p-1}}^T & \\ & & & & (C_N)_{q_{N_p}}^T \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_{N_p-1} \\ \lambda_{N_p} \end{bmatrix} = - \begin{bmatrix} J_{q_1}^T \\ J_{q_2}^T \\ \vdots \\ J_{q_{N_p-1}}^T \\ J_{q_{N_p}}^T \end{bmatrix} \tag{4-16}$$

By using the adjoint equation shown above, the Lagrangian as defined in Equation (4-4), can be defined. Then the gradient can be calculated using Equation (3-9). Notice that on all but the last line, we get multiple $\lambda$'s, *i.e.*, $\lambda_k$ depends on $\lambda_{k+1}$. As a result, the $\Lambda$ stacked matrix should be determined backwards: first, $\lambda_{N_p}$, which is independent of other $\lambda$'s, is calculated, and then used to determine $\lambda_{N_p-1}$, etc:

$$\begin{aligned} (C_{N_p})_{q_{N_p}}^T \lambda_{N_p} &= - J_{q_{N_p}}^T \\ (C_{N_p-1})_{q_{N_p-1}}^T \lambda_{N_p-1} &= - J_{q_{N_p-1}}^T - (C_N)_{q_{N_p-1}}^T \lambda_{N_p} \\ &\vdots \\ (C_1)_{q_1}^T \lambda_1 &= - J_{q_1}^T - (C_2)_{q_1}^T \lambda_2 \end{aligned} \tag{4-17}$$

For this reason, this method is sometimes called the *backwards adjoint* method. Once the Lagrangian is determined, the gradient of $J$ can be calculated in the same way as done in Equation (4-7): $\bigtriangledown_{\tilde{w}} J = J_{\tilde{w}} + \Lambda^T C_{\tilde{w}}$. This results in the following individual derivatives:

$$\begin{aligned} \frac{dJ}{dw_{N_p}} &= \frac{\partial J}{\partial w_{N_p}} + \lambda_{N_p}^T \frac{\partial C_{N_p}}{\partial w_{N_p}} \\ \frac{dJ}{dw_{N_p-1}} &= \frac{\partial J}{\partial w_{N_p-1}} + \lambda_{N_p-1}^T \frac{\partial C_{N_p-1}}{\partial w_{N_p-1}} + \lambda_{N_p}^T \frac{\partial C_{N_p}}{\partial w_{N_p-1}} \\ &\vdots \\ \frac{dJ}{dw_1} &= \frac{\partial J}{\partial w_1} + \lambda_1^T \frac{\partial C_1}{\partial w_1} + \lambda_2^T \frac{\partial C_2}{\partial w_1} \end{aligned} \tag{4-18}$$

As we can see here, the derivative of $J$ with respect to a certain $w_k$, $k \neq N_p$ depends on $\lambda_k$ and $\lambda_{k+1}$, again requiring the backwards calculations. Of course, these equations can also be

solved as matrix equations, but because the matrix $C_k$ is usually already high-dimensional, the total matrix $C$ would become enormous. This would therefore make these calculations computationally too expensive. We therefore solve them seperately in a backwards fashion as shown in Equations (4-17) and (4-18).

## 4-4 Gradient of $J$ for Time-dependent Problems using WFSim

In the previous section, the adjoint method for time-dependent problems was presented. In this section, we will shortly discuss the mathematic expressions of the partial derivatives of $J$ and $C$ that are necessary for the calculations of the gradient $\bigtriangledown_{\tilde{w}} J$ in WFSim. As we have seen in the previous section, we need four partial derivatives to determine the gradient:

1. The partial derivative of cost function $J$ with respect to state $\tilde{q}$, $\frac{dJ}{d\tilde{q}}$;
2. The partial derivative of cost function $J$ with respect to state $\tilde{w}$, $\frac{dJ}{d\tilde{w}}$;
3. The partial derivative of constraint $C$ with respect to state $\tilde{q}$, $\frac{dC}{d\tilde{q}}$;
4. The partial derivative of constraint $C$ with respect to state $\tilde{w}$, $\frac{dC}{d\tilde{w}}$;

The exact expressions of these partial derivatives can be found in Appendix B. Let the objective be to maximize the power production of a wind farm with $N$ turbines over a given prediction horizon $N_p$. This gives the following objective function $J$ using Equation (2-9):

$$J = \sum_{k=1}^{N_p} J_k = -\sum_{k=1}^{N_p} \sum_{n=1}^{N} P_{n,k} = -c_J \sum_{k=1}^{N_p} \sum_{n=1}^{N} \left(U_{n,k}^{r,\perp}\right)^3 \beta_{n,k} \tag{4-19}$$

where $c_J$ is a constant that is independent of input $w$, states $q$ or turbine $n$. Note that $c_J$ contains the rotor area $A_r$ (which we assume is the same for all turbines) and the air density $\rho$ as shown in Equation (2-9). Finally, $\beta_{n,k}$ is the input of turbine $n$ at time instant $k$, as defined in Equation (2-7).

Furthermore we have the constraint equation $C_k$ describing the wind flow at time instant $k$:

$$C_k(q_k, q_{k-1}, w_k) = E(q_{k-1})q_k - Aq_{k-1} - B_c(q_{k-1}) - S_m(q_{k-1}, w_k) \tag{4-20}$$

As shown in Equation (3-8), the total constraint $C$ will be a concatenation of $C_1, C_2, \cdots C_{N_p}$. To use the adjoint method, we now have to find the partial derivatives of $J$ and $C$ with respect to state $\tilde{q}$ and input $\tilde{w}$. First, we recall that the input $w_k$ is restricted to the scaled axial induction factors $\beta$ of all turbines at time $k$. The partial derivative of the objective function $J$ with respect to input $w_k = \begin{bmatrix}\beta_{1,k} & \beta_{2,k} & \dots & \beta_{N,k}\end{bmatrix}^T$, where $N$ is the number of turbines in the farm, can then be found easily:

$$\partial_{w_k} J_k = \frac{\partial J_k}{\partial w_k} = -c_J \begin{bmatrix} (U^{r,\perp}_{n=1})^3 \\ (U^{r,\perp}_{n=2})^3 \\ \vdots \\ (U^{r}_{n=N})^3 \end{bmatrix} \tag{4-21}$$

with $U^{r,\perp}$ defined in Equation (2-22). Concatenating $\partial_{w_1} J_1, \cdots, \partial_{w_{N_p}} J_{N_p}$ as shown in Equation (4-15) then results in the complete partial derivative $\partial_{\tilde{w}} J$.

The partial derivative with respect to the state $q$ is more difficult, since $q$ resides in Equation 4-19 in a nonlinear fashion. Using Equations (2-23) and (4-19), the objective function at time instant $k$, $J_k$, can be written as:

$$J_k = -c_J \sum_{n=1}^{N} \left( \frac{\sum_{r=1}^{m} \sqrt{u_{n,r}^2 + v_{n,r}^2}}{m} \right)^3 \beta_{n,k} \tag{4-22}$$

Now, taking the derivative with respect to the state $q_k$, we get a vector that contains mostly zeros. Only the derivative with respect to velocities $u_{n,r}$ and $v_{n,r}$, which are contained in vector $q_k$, will be nonzero. One instance of such a nonzero entry is given in Equation (4-23):

$$\frac{\partial J_k}{\partial u_{n=1,r=1}} = -\frac{6 c_J}{m} \cdot \frac{u_{n=1,r=1} \cdot \left( \sum_{r=1}^{m} \sqrt{u_{n,r}^2 + v_{n,r}^2} \right)}{\sqrt{u_{n=1,r=1}^2 + v_{n=1,r=1}^2}} \tag{4-23}$$

So $\partial_{q_k} J$ will be an 1 by $n_q$ matrix with mostly zeros and an expression as given in (4-23) on all locations of $u_{n,r}$ and $v_{n,r}$ in $q_k$. The complete partial derivative $\partial_{\tilde{q}} J$ is then a concatenation of $J_1, \cdots, J_{N_p}$. Note that $\partial_{q_{N_p}} J = 0$ since the state $N_p$ influences only the power at time instant $N_p + 1$. This results in the following expression of the complete derivative:

$$\frac{\partial J}{\partial \tilde{q}} = \begin{bmatrix} \partial_{q_1}(J_2) \\ \partial_{q_2}(J_3) \\ \vdots \\ \partial_{q_{N_p-1}}(J_{N_p}) \\ 0 \end{bmatrix} \in \mathbb{R}^{1 \times n_q \cdot N_p} \tag{4-24}$$

Next, we need to determine the derivatives of the constraint given in Equation (4-20). The derivatives of $C$ have already been determined in Section 2-6. $\partial_q C = -\bar{A}$, and $\partial_w C = -\bar{B}$, with $\bar{A}$ and $\bar{B}$ defined in Equation (2-20). Note that, as mentioned, all derivatives and how they are obtained in WFSim can be found in Appendix B.

We therefore now have expressions for all the partial derivatives needed to implement the adjoint method. A typical gradient of $J$ found with the adjoint method for a 3-turbine wind farm is shown in Figure 4-1. In this figure, the input $\tilde{w}$ was kept constant at $\beta = 0.5$ for all turbines over the entire prediction horizon, *i.e.*, greedy control was applied.

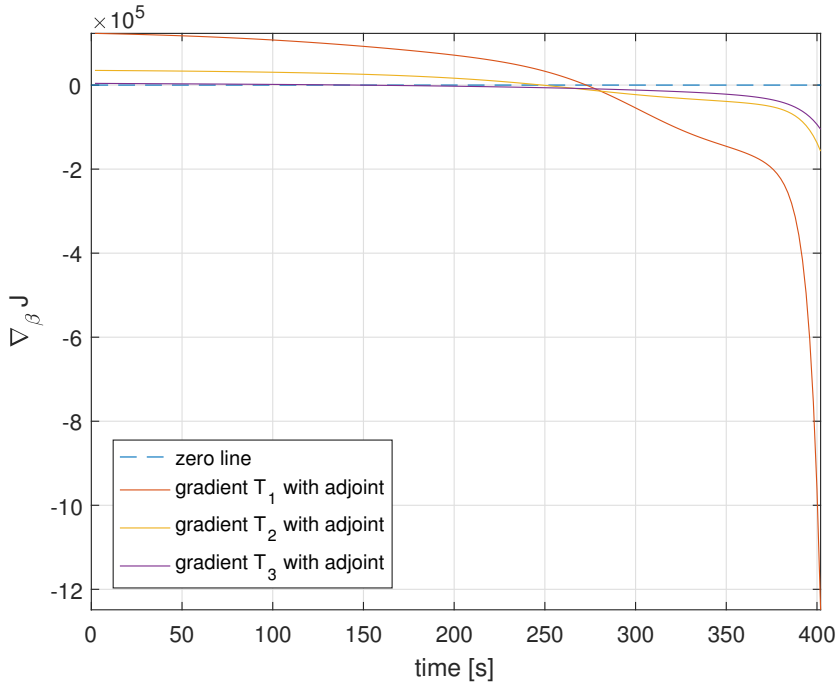



**Figure 4-1:** A typical gradient calculated with the adjoint method.

Note immediately that the gradient is not equal, or even close, to zero for all turbines. Clearly, the greedy control strategy is not optimal. If it were, all gradients would have to be zero as a change in input $w$ would always result in a decrease of power production. Figure 4-1 therefore shows that it is actually useful to do optimization to determine the inputs of the turbines in a 3-turbine wind farm. In the following section, this gradient and its correctness will be discussed.

## 4-5 Gradient Validation using the Numerical Gradient

To check whether the gradient that was obtained using the adjoint method is reliable, the gradient of the cost function $J$ (Equation (4-19)) will be calculated using numerical differentiation.

As seen in Section 4-3, the gradient for time-dependent problems is different than the static gradient. As a result, a gradient that is correct for steady-state calculations, doesn't necessarily has to be correct for the dynamic case as well. Therefore, this case is studied seperately to validate the dynamic gradient. This gradient is obtained by using the adjoint method as described in Section 4-4.

To determine the numerical gradient, the wind flow is first brought to a steady state for a given $\tilde{w} = \begin{bmatrix} \beta_{k=1,n=1} & \dots & \beta_{k=N_p,n=N} \end{bmatrix}^T$, where $N_p$ is the prediction horizon. The numerical gradient can than be computed as:

$$\left(\frac{dJ}{d\beta_{k,n}}\right) = \frac{\bigtriangledown J}{\bigtriangledown \beta_{k,n}} = \frac{J(\beta_{k,n} + \delta) - J(\beta_{k,n})}{\delta} \tag{4-25}$$

By repeating this calculation for different inputs, the numerical gradient over time can be calculated. This gradient can then be compared with the gradient obtained using the adjoint method. Note that calculating the numerical gradient can be a very time-consuming action: to determine the total power production over the prediction horizon for a single excited input, the entire horizon needs to be simulated. If we use the WFSim with a grid of $50 \times 25$ cells and a prediction horizon of $400s$ with a step size of $\Delta t = 2s$, one forward simulation takes around $30s$ on a standard laptop computer. So if we want to calculate every single individual derivative in $\frac{dJ}{d\bar{w}}$, this would take approximately 5 hours. This shows why it is interesting to employ the adjoint method to calculate the gradient.

To mitigate this problem, only one in 10 inputs is excited. This reduces the calculation time of the numerical gradient to approximately 30 minutes, while still providing a gradient that can effectively be compared with the adjoint gradient. Note that this is significantly longer than calculating the adjoint using the adjoint method. This takes approximately 13 seconds for the given prediction horizon.

The adjoint-based gradient and the numerical gradient will be compared using a 3-turbine wind farm. This set-up is particarly interesting since it will be the main set-up used to test the Model Predictive Controller in Chapter 6. The set-up contains three turbines that are placed in line, with a bilateral distance of $7D$, where $D$ is the diameter of the rotors. The dimensions of the wind farm and the turbines can be found in Table 4-1. The grid is given in Figure 4-2(a).

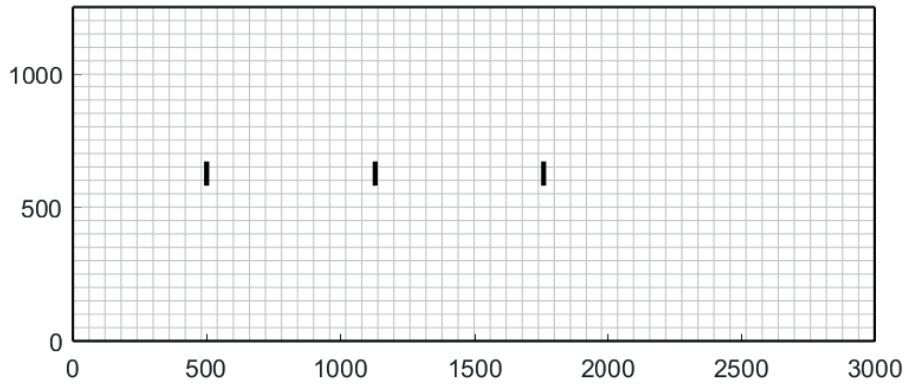
**Table 4-1:** Size of the grid for the 3-turbine wind farm.

| | | |
|---|---|---|
| Number of turbines | $N$ | 3 |
| Rotor diameter [m] | $D$ | 90 |
| Length in x-direction [m] | $L_x$ | 3000 |
| Length in y-direction [m] | $L_y$ | 1250 |
| Locations of rotor center (x) [m] | $R_x$ | $\begin{pmatrix} 500 & 500+7D & 500+14D \end{pmatrix}$ m |
| Locations of rotor center (y) [m] | $R_y$ | $\begin{pmatrix} 0.5L_y & 0.5L_y & 0.5L_y \end{pmatrix}$ |
| Number of grid points (x) | $N_x$ | 50 |
| Number of grid points (y) | $N_y$ | 25 |

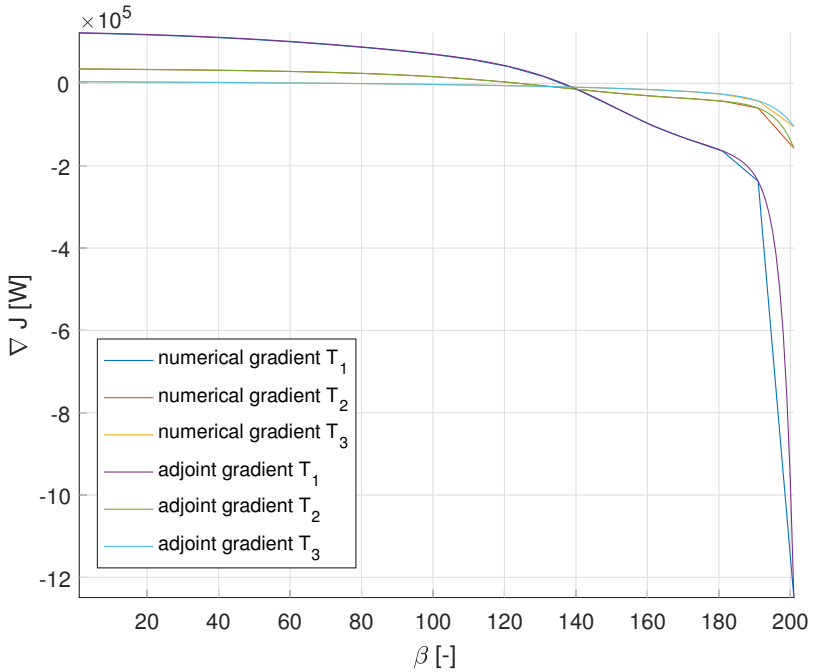
For the inflow velocity, 8 m/s is selected. Initially, input $\beta = 0.5$ over the entire prediction horizon $N_p = 400$ s. This results in the gradients shown in Figure 4-2(b).

If we look purely to the adjoint-based gradients and their numerical counterparts, we notice immediately that these are identical. This indicates that the adjoint method is indeed implemented correctly.

The positive initial values for the gradient of turbines 1 and 2 indicate that for optimal power production, the induction of these turbines should be reduced. Furthermore, reducing the input $\beta_k$ of turbine 1 has the most influence on the power production, since this gradient is much larger. This makes sense intuitively, because derating turbine 1 benefits both downwind




**(a)** Grid in WFSim




**(b)** Gradient

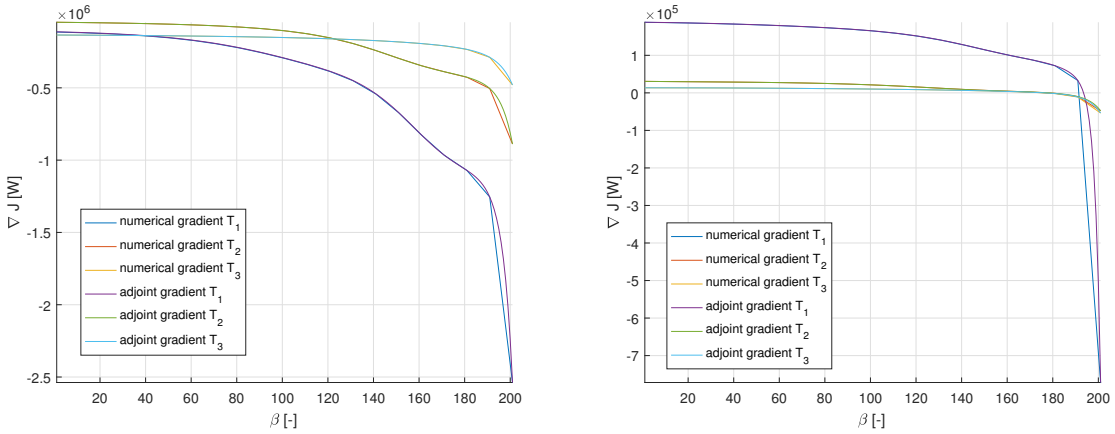
**Figure 4-2:** Verification of the gradient for a 3-turbine case, with $\beta = 0.5$, and the grid used for this simulation.

turbines, whereas derating turbine 2 only benefits turbine 3. Notice that the gradient of the hindmost turbine is again almost zero, indicating that this turbine is close to its optimum.

The shape of the gradient can be explained by studying the set-up of the wind farm. Notice that initially, the gradient of turbine 1 is larger than 0. This implies that the optimal $\beta$ is lower than 0.5 for the first turbine. This makes sense, as a lower $\beta_{n=1}$ would leave more energy in the wind for the second turbine.

The gradient of the third turbine is initially very close to 0, implying that the optimum lies near 0.5. The third turbine has no other turbine in its wake, so its optimum should be the the same as the optimum of a single turbine, which is $\beta = 0.5$ (see Chapter 2).

Between time steps 100 and 150, we notice that the gradient of turbine 1 starts to drop, indicating that the optimal $\beta_k$ should increase. This can be explained by looking at the set-up: the distance between turbines 1 and 2 is $7D = 630$m. With an inflow velocity of 8 m/s and an input of $\beta_k = 0.5$, it takes approximately 200 seconds for the wind to reach turbine 2 after it has passed the first turbine. This means that as soon as we reach the last 200 seconds of the prediction horizon, the wind that passes through the first turbine does not reach the second turbine within the prediction horizon. As a result, in order to achieve maximal power production within the given prediction horizon, the induction of the first turbine can be increased without effecting the power production of the second turbine within said horizon.




**(a)** $\beta = 0.2$ **(b)** $\beta = 0.8$

**Figure 4-3:** Verification of the gradient for $\beta = 0.2$ (a) and $\beta = 0.8$ (b), respectively.

To further verify these results, the adjoint-based gradient is also compared with the numerical version for other values $\beta_k$. These results are shown in Figure 4-3. Here we can see that the numerical gradient again matches the adjoint-based gradient. Also, all gradients are negative for the low value of $\beta_k = 0.2$, indicating an increase of $\beta_k$, whereas the gradients are positive for the high value $\beta_k = 0.8$.

All these results together, over different wind farm configurations and with different input variables, make it reasonable to assume that the adjoint-based gradients are in fact correct. The gradients can therefore be used in an MPC-based control algorithm, which will be discussed in Chapter 6.

# Chapter 5

# The Adjoint-based MPC Implementation

In Chapter 3, a Model Predictive Control (MPC) framework was presented to optimize the power production of a wind farm. Chapter 4 then showed how the gradient of $J$ was determined. In this chapter, the algorithm developed in this thesis will be presented. Section 5-1 will cover the steps taken in the algorithm in order to achieve optimization, while Section 5-2 will elaborate on the line search procedure.

## 5-1 The MPC Algorithm

The implementation of framework given in Figure 3-2 results in Algorithm 2. A few parameters should be defined before the optimization shown in Algorithm 2 can be run: the time step size $h$, the atmospheric conditions $q_{atm}$, the prediction horizon $N_p$, the receding horizon $N_u$ and the number of receding horizons $N_{r,max}$. Furthermore, starting values for $\tilde{w}$ should be defined.

Note that it is important to choose $N_u$ sufficiently small compared to $N_p$, to prevent the descending gradient near time $N_p$, as explained in Chapter 3, to have an effect on the outcome. If $N_u$ is chosen too big, the control inputs will go up near the end of the receding horizon, leading to oscillatory behavior. As mentioned in Chapter 3, a control horizon is not considered, *i.e.*, $N_c = N_p$.

For practical purposes, the control input $\beta$ is bounded. In reality, $\beta$ should always have a value between 0 (*i.e.*, the turbine is inoperative) and 1 (*i.e.*, the turbine completely blocks the wind). To prevent these boundary cases, $\beta \in [0.1 \quad 0.9]$ is chosen. As a starting point, greedy control is applied, *i.e.*, $\beta_{k,n} = 0.5$ for all $k = 1 \dots N_p$, $n = 1 \dots N$.

First, the field is initialized ($q_0$) and the initial cost and gradient is determined. The largest value of this gradient is used as a limit on the step size $\alpha$, *i.e.*, the largest step size allowed is 1 (see Equation (3-6)). Algorithm 2 then takes the following steps in order to optimize the control input $\tilde{w}$:

**Algorithm 2:** Pseudocode for the MPC algorithm

1 $q_0 \leftarrow$ `InitialField` $(q_atm)$; % Compute initial flow fields
2 $\tilde{w} \leftarrow$ `InitialControl`; % Compute initial control sequence
3 $\triangle t \leftarrow h$; % Define sample period
4 $\begin{bmatrix} P(1) & \tilde{q} \end{bmatrix} \leftarrow$ `WFSim` $(q_0, \tilde{w}, q_{atm}, \triangle t, N_p)$ % Forward prediction, Equations (2-18) and (2-23)
5 $J_p \leftarrow$ `GetCost` $(P(1))$; % Determine $J$ based on $P$ (Equation (3-9))
6 $\bigtriangledown J \leftarrow$ `GetAdjoint` $(\tilde{q}, \tilde{w}, P(1))$; % Gradient using adjoint, Equation (4-18)
7 $\alpha_0 \leftarrow \frac{1}{\max(\bigtriangledown J)}$; % Define initial line-search step size
8 $\alpha \leftarrow \alpha_0$;
9 **for** $i = 1 : N_{u,\max}$ % *Time for-loop until final time* $N_u = N_u \cdot N_{u,\max}$
10 $(P(1), \tilde{q}) \leftarrow$ `WFSim` $(q_0(i), \tilde{w}, q_{atm}, \triangle t, N_p)$;
11 $J(1) \leftarrow$ `GetCost` $(P(1))$;
12 $\bigtriangledown J \leftarrow$ `GetAdjoint` $(\tilde{q}, \tilde{w}, P(1))$;
13 $\triangle J \leftarrow \mid \frac{J(1)}{J_p} \mid -1$;
14 $J_p \leftarrow J(1)$;
15 $n \leftarrow 1$;
16 $(\tilde{w}, \tilde{q}, J) \leftarrow$ `LineSearch` $(\tilde{w}, J_p, \epsilon_p)$; % Line search, see Algorithm 3
17 $(P_u(i), q_0(i+1)) \leftarrow$ `WFSim` $(q_0(i), \tilde{w}, \triangle t, N_u)$; % Implementation of first $N_u$ control inputs
18 **end**

1. A forward simulation is done using Wind Farm Simulator (WFSim) to determine the power production for a pre-defined prediction horizon $N_p$ with initial input $\tilde{w}$ (line 10);
2. The gradient $\bigtriangledown_{\tilde{w}} J$ is calculated using the backwards adjoint method (line 12, see Chapter 4);
3. The new input $\tilde{w}^*$ is determined by doing a line search in the search direction $\bigtriangledown_{\tilde{w}} J$ (line 16). For each line search, the algorithm executes a forward simulation of the flow to determine the power production, this will be further elaborated in Section 5-2;
4. A new input $w^*$ is accepted when it results in an increase in power production over $N_p$ (see Section 5-2;
5. Input $w^*$ is implemented over the receding horizon $N_u$ (line 17). We then go back to point 1.

This procedure is followed until the algorithm is no longer able to increase the objective with more than a certain threshold $\epsilon_p$. When this threshold is met, the algorithm will keep the input constant to prevent oscillations around the optimum.

This convergence threshold not only prevents oscillatory behavior when the inputs are close to optimal. It also reduces the computational expense of the algorithm: when the threshold is met, the optimization is terminated. As a result, only one forward simulation still needs to be done. This simulation is done to check that the power production is indeed still close to the previous power production. If this is not the case, the algorithm will again start optimizing the input. This way, the algorithm can deal with significant changes in the atmospheric conditions and adapt to these changes.

## 5-2 The Line-search Algorithm

As mentioned in the previous section, a line-search is done to determine the appropriate step size $\alpha$. The influence of $\alpha$ on the new input $\tilde{w}^*$ is shown in Equation (5-1):

$$\tilde{w}^* = \tilde{w} - \alpha \cdot \bigtriangledown_{\tilde{w}} J \tag{5-1}$$

As mentioned above, the initial value is set at $\alpha = \frac{1}{\max(\bigtriangledown_{\tilde{w}} J)}$, so the initial step size is 1. This is always an overestimation of the optimal step size, as $\beta \in [0.1, 0.9]$. The line-search procedure is shown in Algorithm 3. At each iteration of the line-search, the same horizon is again simulated with the new input $\tilde{w}$ to determine the new power production.

**Algorithm 3:** Pseudocode for the line-search algorithm used in Algorithm 2.

1 $c_1 \leftarrow 0$;
2 $c_2 \leftarrow 0$;
3 **while** $n \leq n_{\max}$ & $(c_1$ & $c_2) \neq 1$ & $\triangle J < \epsilon_p$ % *While-loop for line-search*
4 $\quad \alpha \leftarrow \frac{\alpha}{2}$;
5 $\quad \tilde{w}_{ls}(n) \leftarrow$ `UpdateBeta` $(\tilde{w}, \bigtriangledown J)$; % Equation (5-1)
6 $\quad q_0 \leftarrow$ `LoadState` $(N_u, \tilde{q})$;
7 $\quad P(n+1) \leftarrow$ `WFSim` $(q_0, \tilde{w}_{ls}(n), q_{atm}, \triangle t, N_p)$; % Forward prediction, Equations (2-18) and (2-23)
8 $\quad J(n+1) \leftarrow$ `GetCost` $(P(n+1))$; % Determine $J$ based on $P$ (Equation (3-9))
9 $\quad n_c \leftarrow n+1$;
10 $\quad (c_1, c_2, n_c) \leftarrow$ `StopCriteria` $(J, n)$; % See Equations (5-3) and (**??**)
11 **end**
12 % Reset $\alpha$ if threshold is met
13 **if** $\triangle J < \epsilon_p$ **then**
14 $\quad \alpha \leftarrow \alpha_0$;
15 **end**
16 $\tilde{w} \leftarrow \tilde{w}_{ls}(n_c)$;

In the line-search, we deliberately start with an overestimation of the step size. The step size is then decreased until a satisfying solution is found. A solution is labeled satisfying if it results in a decrease of the cost function $J$ compared to the initial simulation, and a further decrease of the step size does not further decrease $J$.

Notice that the while-algorithm requires the objective $J$ to change more than a certain threshold $\epsilon_p$, as discussed earlier in this chapter. When the objective function no longer increases with more than this threshold, the optimization is stopped and the input $\tilde{w}$ is kept constant.

The stop criteria mentioned on line 10 of Algorithm 3 guarantee that the line-search only terminates when the step size can no longer be improved. Notice in line 3 that both criteria need to be fulfilled to terminate the line-search. Criterion $c_1$ guarantees that the line-search only terminates when the power production exceeds the initial power production:

$$c_1 = \begin{cases} 1 & \text{if } J(n+1) > J(1) \vee J(n) > J(1) \\ 0 & \text{otherwise} \end{cases} \tag{5-2}$$

The second criterion prevents that the line-search terminates when a smaller step could further improve the cost function:

$$c_2 = \begin{cases} 1 & \text{if } J(n+1) < J(n) \\ 0 & \text{otherwise} \end{cases} \tag{5-3}$$

The algorithm is similar to the one presented in [21] in the sense that it also uses the adjoint method to determine the gradient of a cost function combined with a receding-horizon approach. The focus of [21] however is to control turbulent boundary layers using large-eddy simulations. Turbulence therefore plays a large role in this research, whereas in this thesis, turbulence is simplified by tuning the dynamic viscosity $\mu$ (see Chapter 2). In [21], a penalty is included on energy dissipation in the cost function, whereas this thesis focusses on maximizing the power production.

Furthermore, the algorithm of [21] is not intended for real-time control. Consequently, does up to 40 forward simulations for each control window. This is a significant difference with the algorithm presented here, that uses up to 11 (and often only 2 or 3) forward simulations per control window. This results in a significant advantage in computational expense. The shorter computation time of the algorithm presented in this thesis is much more qualified to do on-line dynamic control of a wind farm.

The algorithm developed in this thesis is more closely related to the work done in [22]. The same MPC framework is used and [22] also uses WFSim. It is therefore also interesting to compare the results, as will be done in Chapter 6. The difference between both algorithms lies mostly in the line-search method. [22] uses conjugate gradient methods combined with Armijo backtracking, where it was found here that a simple backtracking algorithm leads to similar results with less computational expense.

Apart from that, [22] also uses projections to set the gradient to zero if the input is at the boundaries that have been set. This is also ommitted here, as it was found to have very little to no effect on the overall control signal implemented. Further comparisons between the results will be discussed in Chapter 6.

The most significant difference however, is the use of the convergence threshold $\epsilon_p$. This threshold stimulates a smooth power production nearby the optimum and prevents redundant or futile simulations. In [22], such a mechanism is not present, resulting in continued simulations even when the power production is no longer improved, as well as a more disturbed power production.

# Chapter 6

# Simulation Results

In this chapter, the results of the nonlinear Model Predictive Control (MPC) discussed in Chapter 5 will be presented. The controller will be tested using three different cases:

1. The atmospheric conditions are constant;
2. The wind velocity is time-varying: midway through the simulations, the wind velocity will increase;
3. The wind direction is time-varying: midway through the simulations, the main flow direction will change.

The objective is the same for each of these three cases: we want to maximize the power production of the wind farms. By changing the atmospheric conditions, we hope to prove that the MPC approach can be used to dynamically optimize a wind farm. The first case is used to answer the first two research subquestions given in Chapter 1: *can we increase the power production of a wind farm with respect to greedy control?* and *can the control algorithm steer the system to an optimal steady-state for constant wind conditions?* The other cases are used to verify the third research subquestion: *can the control algorithm adapt to changing wind conditions?* During each of these three simulation cases, different wind farm lay-outs will be evaluated:

- a 3-turbine wind farm with the turbines aligned with the main flow;
- a 4-turbine wind farm with the turbines aligned with the main flow;
- a 6-turbine wind farm with a 2-by-3 topology.

The layout of these wind farms will be presented in this chapter. Having 3 cases and 3 wind farm topologies, a total of 9 simulation cases has been studied. Sections 6-1 to 6-3 will show one simulation for each of the 3 cases. The other results can be found in Appendix C. This

chapter will show that the controller is able to adapt to changing atmospheric conditions, and is still able to perform better than a greedy controller. In Section 6-4 all the simulation results will be summarized.

Note that the performance of the controller is evaluated using the *instantanious* power production, *i.e.*, the power produced by the wind farm when the optimized control input is implemented. The increase in instantanious power production will be lower than the increase of the objective function $J$, which is typically $> 10\%$. However, the power increase at the end of the prediction horizon, when the wake effects are no longer relevant (see Section 4-5), account for a significant section of this increase. These inputs will not be implemented on the wind farm, and are therefore not considered here.

For these simulations, a couple of variables need to be chosen with care. The prediction horizon $N_p$, receding horizon $N_u$, and the number of cells in x-direction ($N_x$) and y-direction ($N_y$) have significant influence on the performance and computation time of the algorithm. The effect of the number of cells on the computation time is shown in Table 6-1. Here we see the computation time of the forward simulation (with $N_p = 400$ s, and $\Delta t = 2$ s) and the backwards adjoint for different grid densities.

**Table 6-1:** The computation times needed on a standard laptop computer for the forward simulation and backwards adjoint for different grid sizes

| Grid size | Number of cells | Forward simulation | Backwards adjoint |
|---|---|---|---|
| $20 \times 10$ | 200 | 5 s | 1 s |
| $50 \times 25$ | 1250 | 28 s | 12 s |
| $100 \times 50$ | 5000 | 135 s | 54 s |

Table 6-1 shows that the computation time of the forward simulation depents more or less linearly on the number of cells in the grid. The adjoint calculation shows similar behavior, although it increases more than the forward simulation when the number of cells is increased. It is clear that we should look for a grid density that is as low as possible whilst still providing a reliable flow simulation.

Note that, to prevent interference of the boundary conditions on the flow at the turbines, the flow field has to be chosen sufficiently large. To guarantee this, the size of the field is set at 3000 by 1250 m. For the $20 \times 10$ grid given in Table 6-1, this would mean that each cell represents 150 by 125 m, while the rotor diameter is only 90 m. The $20 \times 10$ grid is consequently not considered to be dense enough to give a reliable representation of the interaction between the flow and the turbines . Simulations with the $50 \times 25$ and the $100 \times 50$ grid show very similar results. Therefore, the $50 \times 25$ grid will be used.

To minimize the effect of the decline of the gradient at the end of the prediction horizon on the implemented inputs, the prediction horizon is set at $N_p = 400$ s. The sample time is chosen at $\Delta t = 2$ s, as this provides similar results as $\Delta t = 1$ s while reducing the computation time. As we can see in Table 6-1, one forward simulation takes 28 seconds for these setting. Since each optimization loop requires *at least* 3 forward simulations to determine the step size $\alpha$, this would require the receding horizon to be *at least* $3 \cdot 28 + 12 = 96$ seconds in order for the algorithm to be used real-time. This is however too close to the value of $N_p$. It is therefore not feasible to do real-time control using this algorithm on a laptop computer. To achieve real-time control, more computational power is required for the optimization to be

done sufficiently fast. As a trade-off between computation time and simulation results, we thus set the receding horizon at $N_u = 20$ s.

## 6-1 Case 1: Constant Atmospheric Conditions

In the first case, the atmospheric conditions are assumed to be constant. As a result, the wind velocity at the boundaries will be constant over the entire simulation. The flow and control settings used for these simulations are given in Table 6-2.

**Table 6-2:** The flow and control settings used in case 1.

| | | |
|---|---|---|
| Inflow velocity (x-direction) | $u_b$ | 8m/s |
| Inflow velocity (y-direction) | $v_b$ | 0m/s |
| Initial pressure | $p_0$ | 0Pa |
| Time step size | $\Delta t$ | 2s |
| Prediction horizon | $N_p$ | 400s |
| Receding horizon | $N_u$ | 20s |
| Number of receding horizons | $N_{u,\max}$ | 30 |
| Maximum number of line searches | $i_{\max}$ | 10 |
| Threshold of the cost function | $\epsilon_p$ | 0.002 |

Note that the total simulation time for this case will be 600 seconds, since 30 windows of 20 seconds are implemented. Since $u$ is the velocity aligned with the x-axis and $v_b = 0$, the flow is aligned with the rows of wind turbines (see Figure 6-1). For this situation, we would ideally want the system to go to an optimal steady-state. However, theoretically, this will only be the case if $N_c < \infty$ and $N_p \to \infty$.

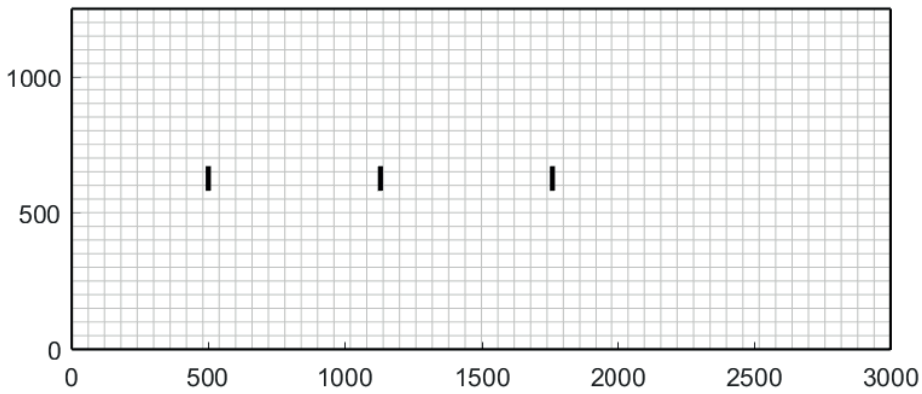



**Figure 6-1:** Representation of the grid with 3 turbines in WFSim.

For this simulation case, we will evaluate the response of the system for the 3-turbine layout. Table 6-3 shows the dimensions of this wind farm layout while Figure 6-1 depicts the resulting grid including the wind turbine locations.

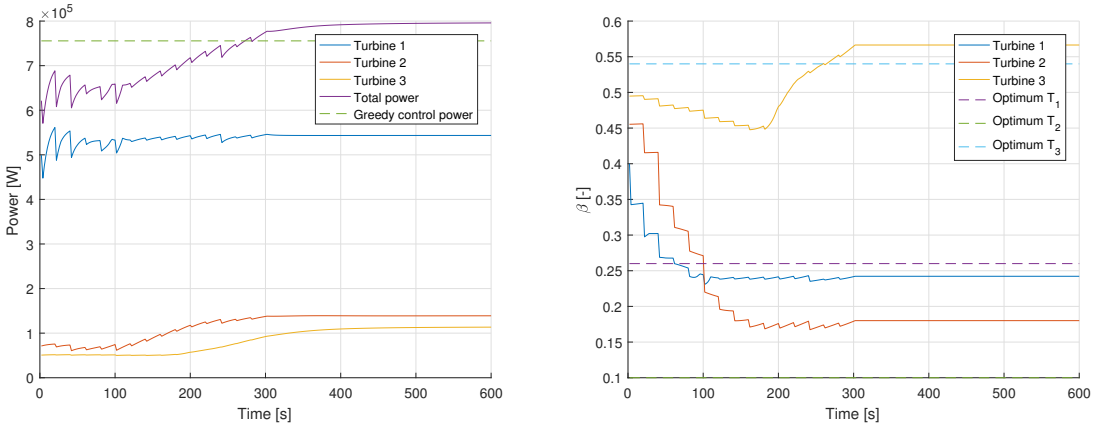
By doing an extensive grid search, the optimum of the 3-turbine wind farm for constant atmospheric conditions can be found. This was done by letting the model go to steady-state

**Table 6-3:** Size of the grid for the 3-turbine wind farm.

| | | |
|---|---|---|
| Number of turbines | $N$ | 3 |
| Rotor diameter [m] | $D$ | 90 |
| Length in x-direction [m] | $L_x$ | 3000 |
| Length in y-direction [m] | $L_y$ | 1250 |
| Locations of rotor center ($x$) [m] | $R_x$ | $\begin{pmatrix} 500 & 500+7D & 500+14D \end{pmatrix}$ |
| Locations of rotor center ($y$) [m] | $R_y$ | $\begin{pmatrix} 0.5L_y & 0.5L_y & 0.5L_y \end{pmatrix}$ |
| Number of grid points ($x$) | $N_x$ | 50 |
| Number of grid points ($y$) | $N_y$ | 25 |

for all different input values within the given boundaries. The input that results in the highest power output over the prediction horizon $N_p = 400s$ is considered the optimum. For these conditions, the optimal input in steady-state equals $\beta = \begin{bmatrix} 0.25 & 0.10 & 0.54 \end{bmatrix}$.

This optimum makes sense if we look at the layout in Figure 6-1: the first and second turbine are derated so the third turbine can produce more power. As discussed in Chapter 3, the dynamic controller will not necessarily steer the inputs towards the optimal steady-state. Instead, we would expect the input to increase as we approach the end of a prediction horizon. The steady-state result can however be used to evaluate the performance of the controller.




**(a)** Power **(b)** Input

**Figure 6-2:** The power production (a) and corresponding inputs (b) of a 3-turbine WF with $\epsilon_p = 0.002$.

For this case, $\epsilon_p = 0.002$, *i.e.*, the optimization will be terminated when the objective function increases with less than 0.2%. It was found that for this value of $\epsilon_p$, the optimization terminates when further optimization is no longer useful. The effect of including this parameter in the MPC algorithm will be shown for the 3-turbine layout.

Using Algorithm 2, the power over $N_p$ will be optimized and the control inputs found will be implemented over receding horizon $N_u$. This results in the response shown in Figure 6-2. We can see in Figure 6-2(a) that the power production of the optimized controller quickly exceeds the power production of the greedy controller. After approximately 280 seconds (so

14 implementations of calculated inputs), the instantanious power of the turbines exceeds the greedy control. As we can see in Figure 6-2(b), which gives the implemented control inputs over time, the terminal constraint $\epsilon_p$ is met after 320 seconds, resulting in a constant input from then on. The final power production is 5.8% better than the greedy control strategy.

Notice that even though it is derated at $\beta = 0.24$, turbine 1 still produces by far the most power of the three turbines. Here we can clearly see the effect that the wake of a turbine has on the power production of downwind turbines. Observe that the final control inputs in Figure 6-2 are very close to the steady-state optimum $\beta = \begin{bmatrix} 0.25 & 0.10 & 0.54 \end{bmatrix}$ found by doing a grid search.

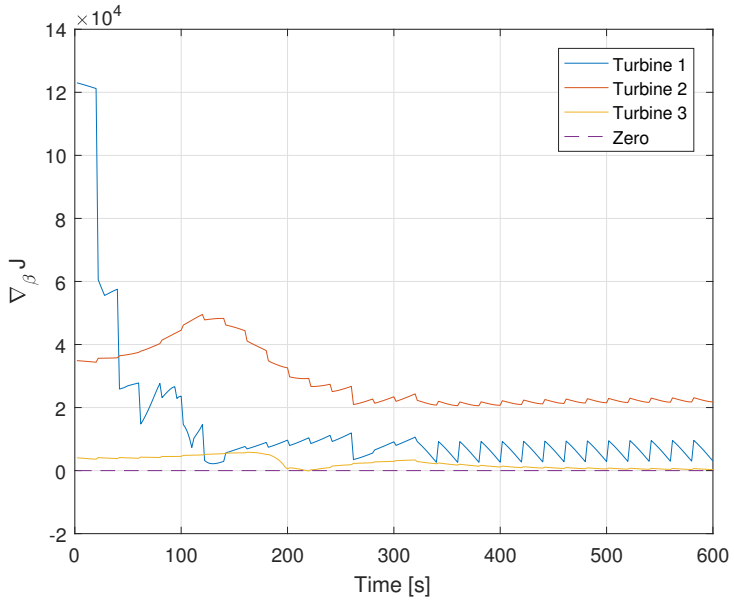


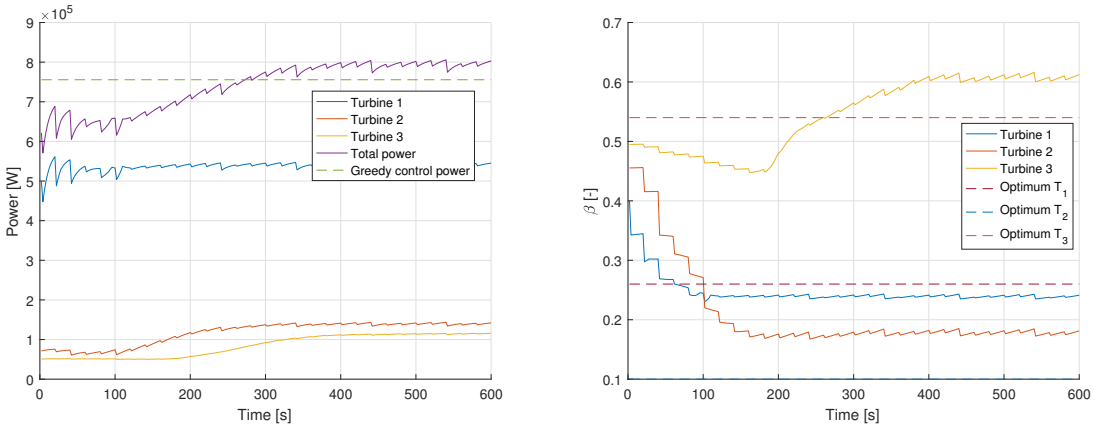

**Figure 6-3:** The gradient belonging to the results of Figure 6-2.

If we look at Figure 6-3, we can see that the gradient still gives the correct direction for the input to go towards the steady-state optimum. The values of the gradients are however relatively small, and further optimization is therefore useless (as can be seen in Figure 6-4). This is why the optimization is terminated by using the convergence threshold $\epsilon_p$.

Once the power production increases with less than $\epsilon_p$, the optimization is stopped and the current input is kept constant. Note that it is important to chose $\epsilon_p$ with deliberation. Choosing it too large will mean that the algorithm terminates the optimization prematurely, while choosing it too small will result in an algorithm that never converges.

To demonstrate the effect that this convergence threshold $\epsilon_p$ has on the response, we repeat the experiment of the previous section. The only difference with the previous simulation is that this threshold is not used now, *i.e.*, $\epsilon_p = 0$. Figure 6-4 shows the result of this simulation.

We can see here that the power production increases in the first 300s of the simulation, but when it comes close to its optimum, the power keeps fluctuating. This can be explained if we look at the input in Figure 6-4(b): the input increases over each receding horizon, but then drops at the beginning of the next horizon. As a result, the power production continues to exhibit this oscillatory behavior.




**(a)** Power **(b)** Input

**Figure 6-4:** The power production (a) and corresponding inputs (b) of a 3-turbine WF without threshold $\epsilon_p$.

Figure 6-2 shows the results of the with $\epsilon_p = 0.002$. The results are equivalent for the first 300s of the simulation, but now the inputs become constant when the oscillations used to start. As a result, also the power production exhibits much smoother behavior.

## 6-2 Case 2: Changing Inflow Wind Velocity

In this section, we will discuss the case where the inflow velocity is no longer constant. The conditions as well as the control parameters are shown in Table 6-4. We will only evaluate the 4-turbine wind farm layout. The other wind farms produce similar results and these results can be found in Appendix C.

**Table 6-4:** The flow and control settings used in case 2.

| | | |
|---|---|---|
| Inflow velocity (x-direction) | $u_b$ | $\begin{cases} 8\text{m/s} & \text{if } i \leq 0.5N_{u,\max} \\ 10\text{m/s} & \text{if } i > 0.5N_{u,\max} \end{cases}$ |
| Inflow velocity (y-direction) | $v_b$ | 0m/s |
| Initial pressure | $p_0$ | 0Pa |
| Time step size | $\Delta t$ | 2s |
| Prediction horizon | $N_p$ | 400s |
| Receding horizon | $N_u$ | 20s |
| Number of receding horizons | $N_{u,\max}$ | 60 |
| Maximum number of line searches | $i_{\max}$ | 10 |
| Threshold of the cost function | $\epsilon_p$ | 0.002 |

Note that the simulation time is now 1200 seconds, as opposed to 600 seconds in case 1. Note furthermore that although we have full state knowledge, we do not assume knowledge of changes in the atmospheric conditions. Consequently, the algorithm keeps these conditions constant over its prediction horizon. In other words, the controller does not know whether the

conditions will change within its prediction horizon, and consequently assumes they remain constant. It therefore does not anticipate on, *e.g.*, a changing inflow velocity.

As mentioned, this case will be evaluated for a 4-turbine WF layout. The dimensions of this wind farm are given in Table 6-5.

**Table 6-5:** Size of the grid for the 4-turbine wind farm.

| | | |
|---|---|---|
| Number of turbines | $N$ | 4 |
| Rotor diameter [m] | $D$ | 90 |
| Length in x-direction [m] | $L_x$ | 3000 |
| Length in y-direction [m] | $L_y$ | 1250 |
| Locations of rotor center ($x$) [m] | $R_x$ | $\begin{pmatrix} 500 & 500+7D & 500+14D & 500+21D \end{pmatrix}$ |
| Locations of rotor center ($y$) [m] | $R_y$ | $\begin{pmatrix} 0.5L_y & 0.5L_y & 0.5L_y & 0.5L_y \end{pmatrix}$ |
| Number of grid points ($x$) | $N_x$ | 50 |
| Number of grid points ($y$) | $N_y$ | 25 |

Notice that this wind farm contains 4 turbines aligned with the dominant wind direction. Figure 6-5 gives a schematic representation of this wind farm topology in WFSim.

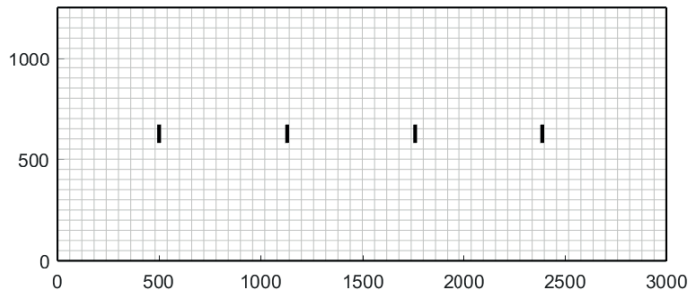


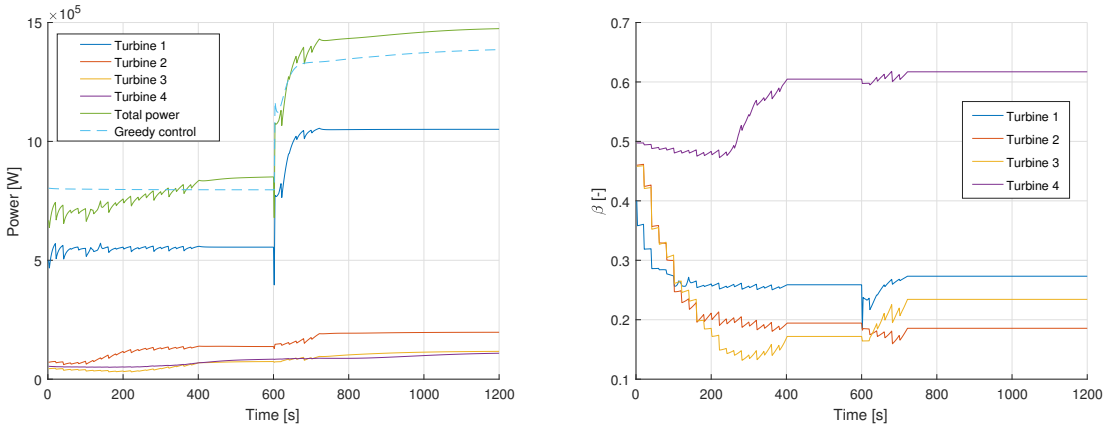

**Figure 6-5:** Representation of the grid with 4 turbines in WFSim.

Figure 6-6 shows the results for these changing conditions on a 4-turbine wind farm. Notice that the algorithm does not anticipate on the changing atmospheric conditions: the input $\beta$ remains constant until $t = 600$s, when the change in wind velocity occurs. Notice that after the velocity changes to $u = 10$m/s, the greedy control strategy is immediately outperformed by the controller after one receding horizon of 20 seconds. After 120 seconds, the threshold $\epsilon_p$ is met and steady-state is reached. This can be explained by looking at Figure 6-6(b): the steady-state values of input $\beta$ do not differ much for both inflow velocities.

With the increased wind velocity, the relative power gain compared to greedy control is slightly decreased: the controller now yiels an instantanious power increase of 6.4%.

## 6-3 Case 3: Changing Wind Direction

In this section, we will investigate a different change in the atmospheric conditions: a change in longitudinal and lateral flow velocities at the boundaries. The initial conditions and control parameters can be found in Table 6-6. In this section, only the 6-turbine wind farm will be




**(a)** Power **(b)** Input

**Figure 6-6:** The power production (a) and corresponding inputs (b) of a 4-turbine WF with changing inflow velocity.

discussed. Like in the previous cases, the other wind farms exhibit similar behavior and can be found in Appendix C.

**Table 6-6:** The flow and control settings used in case 3.

| | | |
|---|---|---|
| Inflow velocity (x-direction) | $u_b$ | $\begin{cases} 8\text{m/s} & \text{if } i \leq 0.5N_{u,\max} \\ 8 \cdot \cos(30°)\text{m/s} & \text{if } i > 0.5N_{u,\max} \end{cases}$ |
| Inflow velocity (y-direction) | $v_b$ | $\begin{cases} 0\text{m/s} & \text{if } i \leq 0.5N_{u,\max} \\ 8 \cdot \sin(30°)\text{m/s} & \text{if } i > 0.5N_{u,\max} \end{cases}$ |
| Initial pressure | $p_0$ | 0Pa |
| Time step size | $\Delta t$ | 2s |
| Prediction horizon | $N_p$ | 400s |
| Receding horizon | $N_u$ | 20s |
| Number of receding horizons | $N_{u,\max}$ | 60 |
| Maximum number of line searches | $i_{\max}$ | 10 |
| Threshold of the cost function | $\epsilon_p$ | 0.002 |

Note that although the inflow angle $\phi$ changes, the only input variable is still $\beta$: the yaw angle is out of the scope of this thesis. As a result, the wind turbines do not align with the wind direction as it changes. We therefore expect that the power production will decrease after the wind direction changes, since the perpendicular wind velocity $U^{r,\perp}$ will decrease.

Furthermore, we would expect that the controller will yield a smaller increase in power production compared to the greedy strategy, since the turbines will now only have partial wake interaction. Note that without wake interaction, greedy control is already the optimal control strategy and optimization is therefore unnecessary. With an inflow angle of 30°, wake interaction will however still play a role.

The 6-turbine wind farm layout evaluated in this section has a 2 by 3 topology, as shown in Figure 6-7. The dimensions of this wind farm are given in Table 6-7. Figure 6-8 shows the

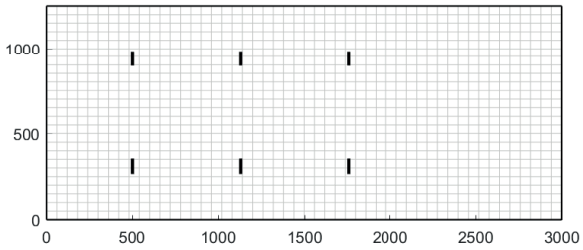
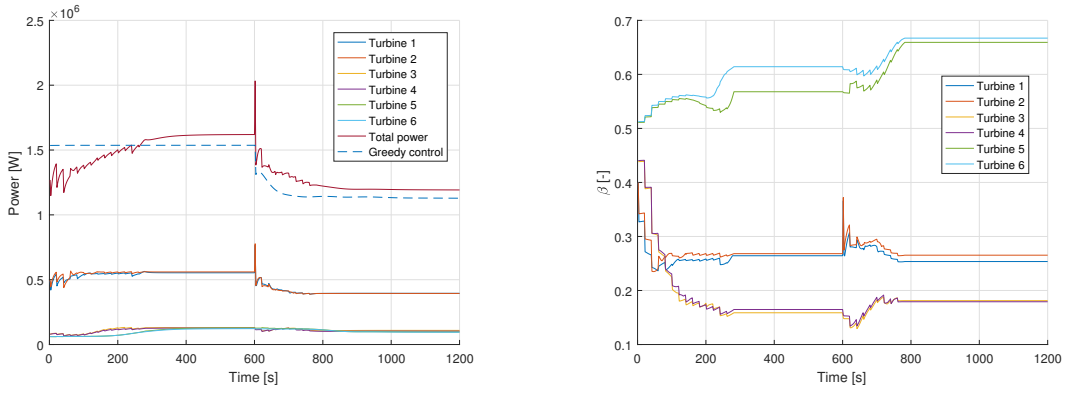



**Figure 6-7:** Representation of the grid with 6 turbines in WFSim.

**Table 6-7:** Size of the grid for the 4-turbine wind farm.

| | | |
|---|---|---|
| Number of turbines | $N$ | 6 |
| Rotor diameter [m] | $D$ | 90 |
| Length in x-direction [m] | $L_x$ | 3000 |
| Length in y-direction [m] | $L_y$ | 1250 |
| Locations of rotor center ($x$) [m] | $R_x$ | $\begin{pmatrix} 500 & 500 & 500+7D & 500+7D & 500+14D & 500+14D \end{pmatrix}$ |
| Locations of rotor center ($y$) [m] | $R_y$ | $3 \times \begin{pmatrix} 0.5L_y - 3.5D & 0.5L_y + 3.5D \end{pmatrix}$ |
| Number of grid points ($x$) | $N_x$ | 50 |
| Number of grid points ($y$) | $N_y$ | 25 |

results obtained for this wind farm.




**(a)** Power **(b)** Input

**Figure 6-8:** The power production (a) and corresponding inputs (b) of a 6-turbine WF.

As expected, the power production decreases after the longitudinal and lateral wind velocities change. The MPC algorithm quickly adapts to the new conditions, and already in the first receding horizon, the power production is higher than for the greedy control strategy.

Threshold $\epsilon_p$ is met after only 9 receding horizons, *i.e.*, 180 seconds. From this moment on, the power production is again constant. In this steady-state, the controller yields 5.5% more power than the greedy control stategy.

## 6-4 Summary

In this section, we will shortly summarize the results obtained for the three cases using the three wind farm layouts. As mentioned before, the results that are not shown in this chapter, can be found in Appendix C. The performance of the controller with respect to the greedy control is shown in Table 6-8. Notice that in all cases, the controller outperforms the conventionally used greedy control strategy.

**Table 6-8:** Performance of the optimization algorithm with respect to the greedy control stategy.

| Cases | 3T | 4T | 6T |
|---|---|---|---|
| **1: Constant conditions** | +5.8% | +7.0% | +5.5% |
| **2: Changing velocity** | +6.4% | +5.3% | +5.3% |
| **3: Changing direction** | +5.4% | +6.4% | +5.5% |

Clearly, the algorithm succeeds in its goal to increase the power production of a wind farm in WFSim. Regardless of the different cases, the controller performs between 5.3% and 7% better than greedy control. It is able to adapt to changing wind conditions and will reach an equilibrium in a matter of minutes.

# Chapter 7

# Conclusions

In this thesis, a nonlinear Model Predictive Control (MPC) framework has been discussed that optimizes the power production of a wind farm. The wind farm model used was the Wind Farm Simulator (WFSim), a 2-dimensional medium-fidelity wind farm model that was developed for control purposes. To determine the optimal control inputs of each individual over time, the adjoint method was used to provide the derivative of the objective function $J$ with respect to the control inputs. In this thesis, the control inputs are the (scaled) axial induction factors of the turbines over the prediction horizon $N_p$.

The algorithm that has been presented in this thesis is unique in a few ways. First of all, by using the WFSim model and an efficient line-search algorithm, it is computationally inexpensive compared to other similar algorithms developed, such as [21]. Secondly, it is a dynamic controller that has been proven to be able to cope with changing atmospheric conditions. As such, combined with the relatively fast computation time, this controller could possibly be implemented on-line to maximize the power production of existing wind farms.

Furthermore, a convergence threshold $\epsilon_p$ has been introduced in this thesis that significantly improves the behavior of the controller when it is close to the optimum. With this convergence threshold, redundant forward simulations that provide no or very little improvement of the cost function are prevented. More importantly, this threshold prevents oscillatory behavior of the input variables in the close-to-optimal region and ensures a smooth power production in said region.

This thesis has subsequently presented a solution to the complicated nonlinear economic MPC problem defined as the maximization of the power production of a wind farm. For different wind farm layouts and in different, changing, atmospheric condition, the controller developed in this thesis has consistently outperformed the conventional greedy control strategy. For all cases, the power production was increased by at least 5.3%, and even a increase of 7.0% has been observed.

## 7-1 Answers to the Research Questions

In Chapter 1, a research objective has been presented, as well as 4 subquestions. In this section, these research objectives will be answered. These subobjectives will first be answered:

1. **Can we increase the power production of a wind farm with respect to greedy control, using only the Axial Induction Factor (AIF) of the turbines as input, under constant wind conditions?** We have seen in Section 6-1 that the power production of a 3-turbine wind farm is increased with respect to greedy control using Axial Induction Control (AIC). Appendix C furthermore shows that the same is the case for a 4-turbine and a 6-turbine wind farm. We therefore conclude that we can in fact increase the power production of a wind farm with respect to greedy control. Answer: **yes**.

2. **Can the control algorithm steer the system to an optimal steady-state for constant wind conditions?** In Section 6-1, the convergence threshold $\epsilon_p$ has been discussed. With this threshold, we are able to obtain steady-state power production for constant wind conditions. Answer: **yes**.

3. **Can the control algorithm adapt to changing wind conditions?** As we have seen in Sections 6-2 and 6-3, the algorithm adapts very quickly to changing wind conditions. For both cases, the power production of the wind farms still exceeds the power produced using greedy control. We thus conclude that the algorithm is in fact able to adapt to changing wind conditions. Answer: **yes**

4. **Can the control input be determined fast enough on a standard laptop computer for real-time dynamic control to be possible?** In Chapter 6, we discussed the computation time needed to obtain a reliable forward simulation for a sufficiently large prediction horizon $N_p$. The computation time needed to provide this simulation on a laptop computer was 28 seconds. Since the algorithm requires at least 3 forward simulations per optimization, we concluded that a receding horizon of at least 96 seconds was required, which is too close to the prediction horizon of $N_p = 400$ s. We consequently conclude that the control input can not be calculated fast enough to enable real-time control. Answer: **no**.

Note that this thesis provides a solution for 3 of the 4 formulated research subobjectives. Although the fourth subobjective is not fulfilled, it does not mean that real-time control is not possible with the algorithm developed in this thesis. The computational power used for these simulations was very limited. Given more computational power, this objective might still be feasible. This leaves us the main research objective, which was defined as:

**With the proposed model-based framework, can we develop a dynamic closed-loop control algorithm that optimizes the power production of a wind farm?**

As we have seen in Chapter 6, the closed-loop control algorithm that has been developed dynamically maximizes the power production for different wind farm layouts. It is also able to adapt to different atmospheric conditions. Although the steady-state optimum is not reached exactly, a dynamic optimum is reached using the convergence threshold introduced in Chapter 5. Accordingly, we can conclude that the main research objective was accomplished.

## 7-2 Future Work

Although this thesis presents solutions to currently existing dynamic wind farm control problems, there is still research to be done in the field of nonlinear MPC for wind farms. In this section, we will shortly discuss the research that could be done in the future to further improve the results that were presented here. The future work can be summarized by these three points:

- Include the yaw angle as a control variable;
- Apply MPC for different objective functions, *e.g.* minimizing the forces on turbines or following a reference power signal;
- Implement the MPC algorithm on a real wind farm or, alternatively, a high-fidelity model, augmented with a Kalman filter to estimate the states;
- Study the feasibility to use the algorithm on-line.

First of all, yaw control was not considered in this thesis. As a result, wake redirection control, where the wake is steered away from downwind turbines, is not possible. WFSim does allow the yaw angle as a control variable. It is therefore possible to include this in the algorithm presented in this thesis. This would of course mean an extension of the control variable $\tilde{w}$. Subsequently, new calculations would need to be made to determine the partial derivatives needed for the adjoint method. The mathematical expressions needed to implement this have been provided in this thesis.

Secondly, different objective functions could be implemented in this MPC framework: the functionality is not limited to power maximization. It could also be used to minimize the forces or fatigue stresses on the turbines. The forces on the turbines are also provided in WFSim, so all requirements to implement this are available. Another objective that could be implemented, is Active Power Control (APC). In APC, the goal is to follow a reference power signal over time. For such an objective, the cost function can be written as a quadratic function. As a result, classical MPC can be implemented as opposed to economic MPC.

Furthermore, we have used WFSim as both the model and the actual wind farm. If the framework were to be implemented on a real wind farm (or, alternatively, a high-fidelity wind farm model), it is possible that the controller performance would decrease. However, since this has not been studied in this thesis, it is hard to make any predictions about this. It would therefore be very useful to implement the work done in this thesis on a real wind farm or high-fidelity wind farm model. This would of course present some new problems, since we now assumed full state knowledge. However, in a real wind farm, is it not feasible to measure the wind velocity at every point in the total farm. As a result, a Kalman filter might be necessary to make an estimation of the states that can be used by the optimizer to determine the optimal input.

Finally, the implementability of this thesis to do on-line control on a real wind farm has not been investigated. It would be interesting to investigate the computational power necessary to optimize the control inputs within the time of one receding horizon. If it were possible to do this, the algorithm could determine the control input for the next receding horizon while the current input is implemented.

# Appendix A

# Coefficients of the fully discretized Navier-Stokes equations

Fully discretizing the Navier-Stokes equations resulted in the following equations:

$$a^{px}_{i,J}u_{i,J} = \begin{pmatrix} a^{nx}_{i,J} & a^{sx}_{i,J} & a^{wx}_{i,J} & a^{ex}_{i,J} \end{pmatrix} \begin{pmatrix} u_{i,J+1} \\ u_{i,J-1} \\ u_{i-1,J} \\ u_{i+1,J} \end{pmatrix} - (p_{I,J} - p_{I-1,J})\,\delta y_{j,j+1} + f^{x}_{i,J} \quad \text{(x-momentum)}$$
$$\forall i = I \in \{3, N_x - 1\}, j = J \in \{2, N_y - 1\},$$

$$a^{py}_{I,j}v_{I,j} = \begin{pmatrix} a^{ny}_{I,j} & a^{sy}_{I,j} & a^{wy}_{I,j} & a^{ey}_{I,j} \end{pmatrix} \begin{pmatrix} v_{I,j+1} \\ v_{I,j-1} \\ v_{I-1,j} \\ v_{I+1,j} \end{pmatrix} - (p_{I,J} - p_{I,J-1})\,\delta x_{i,i+1} + f^{y}_{I,j} \quad \text{(y-momentum)}$$
$$\forall i = I \in \{2, N_x - 1\}, j = J \in \{3, N_y - 1\},$$

$$0 = \delta y_{j,j+1}\,(u_{i+1,J} - u_{i,J}) + \delta x_{i,i+1}\,(v_{I,j+1} - v_{I,j}) \quad \text{(continuity)}$$
$$\forall i = I \in \{2, N_x - 1\}, j = J \in \{2, N_y - 1\}.$$

The coefficients $a^{\bullet}_{\bullet,\bullet}$ are defined in Table A-1.

**Table A-1:** Coefficients in the fully discretized Navier-Stokes equations.

$$a_{i,J}^{wx} = \max\left[F_{i,J}^{wx}, \left(D_{i,J}^{wx} + \frac{F_{i,J}^{wx}}{2}\right), 0\right] \qquad a_{i,J}^{sx} = \max\left[F_{i,J}^{sx}, \left(D_{i,J}^{sx} + \frac{F_{i,J}^{sx}}{2}\right), 0\right] + T_{i,J}^{sx}$$
$$a_{i,J}^{ex} = \max\left[-F_{i,J}^{ex}, \left(D_{i,J}^{ex} - \frac{F_{i,J}^{ex}}{2}\right), 0\right] \qquad a_{i,J}^{nx} = \max\left[-F_{i,J}^{nx}, \left(D_{i,J}^{nx} - \frac{F_{i,J}^{nx}}{2}\right), 0\right] + T_{i,J}^{nx}$$

$$a_{i,J}^{px} = a_{i,J}^{nx} + a_{i,J}^{ex} + a_{i,J}^{sx} + a_{i,J}^{wx} + F_{i,J}^{nx} + F_{i,J}^{ex} - F_{i,J}^{sx} - F_{i,J}^{wx} + T_{i,J}^{px} + a_0^{px}$$

$$a_{I,j}^{wy} = \max\left[F_{I,j}^{wy}, \left(D_{I,j}^{wy} + \frac{F_{I,j}^{wy}}{2}\right), 0\right] \qquad a_{I,j}^{sy} = \max\left[F_{I,j}^{sy}, \left(D_{I,j}^{sy} + \frac{F_{I,j}^{sy}}{2}\right), 0\right]$$
$$a_{I,j}^{ey} = \max\left[-F_{I,j}^{ey}, \left(D_{I,j}^{ey} - \frac{F_{I,j}^{ey}}{2}\right), 0\right] \qquad a_{I,j}^{ny} = \max\left[-F_{I,j}^{ny}, \left(D_{I,j}^{ny} - \frac{F_{I,j}^{ny}}{2}\right), 0\right]$$

$$a_{I,j}^{py} = a_{I,j}^{ny} + a_{I,j}^{ey} + a_{I,j}^{sy} + a_{I,j}^{wy} + F_{I,j}^{ny} + F_{I,j}^{ey} - F_{I,j}^{sy} - F_{I,j}^{wy} + a_0^{py}$$

in which:

$$F_{i,J}^{wx} = \tfrac{1}{2}\rho\left(u_{i,J} + u_{i-1,J}\right)\delta y_{j,j+1} \qquad D_{i,J}^{wx} = \mu\frac{\delta y_{j,j+1}}{\Delta x_{I-1,I}} \qquad a_0^{px} = \frac{\Delta x_{I-1,I}\delta y_{j,j+1}}{\Delta t}$$
$$F_{i,J}^{ex} = \tfrac{1}{2}\rho\left(u_{i+1,J} + u_{i,J}\right)\delta y_{j,j+1} \qquad D_{i,J}^{ex} = \mu\frac{\delta y_{j,j+1}}{\Delta x_{I-1,I}} \qquad \Delta x_{I-1,I} = x_I - x_{I-1}$$
$$F_{i,J}^{sx} = \tfrac{1}{2}\rho\left(v_{I,j} + v_{I-1,j}\right)\Delta x_{I-1,I} \qquad D_{i,J}^{sx} = \mu\frac{\Delta x_{I-1,I}}{\delta y_{j,j+1}} \qquad \delta y_{j,j+1} = y_{j+1} - y_j$$
$$F_{i,J}^{nx} = \tfrac{1}{2}\rho\left(v_{I,j+1} + v_{I-1,j+1}\right)\Delta x_{I-1,I} \qquad D_{i,J}^{nx} = \mu\frac{\Delta x_{I-1,I}}{\delta y_{j,j+1}}$$
$$F_{I,j}^{wy} = \tfrac{1}{2}\rho\left(u_{i,J} + u_{i,J-1}\right)\Delta y_{J-1,J} \qquad D_{I,j}^{wy} = \mu\frac{\Delta y_{J-1,J}}{\delta x_{i,i+1}} \qquad a_0^{py} = \frac{\Delta y_{J-1,J}\delta x_{i,i+1}}{\Delta t}$$
$$F_{I,j}^{ey} = \tfrac{1}{2}\rho\left(u_{i+1,J} + u_{i+1,J-1}\right)\Delta y_{J-1,J} \qquad D_{I,j}^{ey} = \mu\frac{\Delta y_{J-1,J}}{\delta x_{i,i+1}} \qquad \Delta y_{J-1,J} = y_J - y_{J-1}$$
$$F_{I,j}^{sy} = \tfrac{1}{2}\rho\left(v_{I,j-1} + v_{I,j}\right)\delta x_{i,i+1} \qquad D_{I,j}^{sy} = \mu\frac{\delta x_{i,i+1}}{\Delta y_{J-1,J}} \qquad \delta x_{i,i+1} = x_{i+1} - x_i$$
$$F_{I,j}^{ny} = \tfrac{1}{2}\rho\left(v_{I,j} + v_{I,j+1}\right)\delta x_{i,i+1} \qquad D_{I,j}^{ny} = \mu\frac{\delta x_{i,i+1}}{\Delta y_{J-1,J}}$$

$$T_{i,J}^{nx} = \frac{\rho l_u^2 \Delta x_{I-1,I}}{(\Delta y_{J,J+1})^2}\left|u_{i,J+1} - u_{i,J}\right|, \qquad T_{i,J}^{sx} = \frac{\rho l_u^2 \Delta x_{I-1,I}}{(\Delta y_{J-1,J})^2}\left|u_{i,J} - u_{i,J-1}\right|, \qquad T_{i,J}^{px} = T_{i,J}^{sx} + T_{i,J}^{nx}$$

$a^{wx}(u_{i-1,J}, u_{i,J})$, $a^{ex}(u_{i,J}, u_{i+1,J})$,
$a^{nx}(v_{I-1,j+1}, v_{I,j+1}, u_{i,J}, u_{i,J+1})$, $a^{sx}(v_{I-1,j}, v_{I,j}, u_{i,J-1}, u_{i,J})$,
$a^{px}(u_{i-1,J}, u_{i,J-1}, u_{i,J}, u_{i,J+1}, u_{i+1,J}, v_{I-1,j}, v_{I-1,j+1}, v_{I,j}, v_{I,j+1})$,
$a^{wy}(u_{i,J-1}, u_{i,J})$, $a^{ey}(u_{i+1,J-1}, u_{i+1,J})$, $a^{ny}(v_{I,j+1}, v_{I,j})$, $a^{sy}(v_{I,j}, v_{I,j-1})$,
$a^{py}(u_{i,J-1}, u_{i,J}, u_{i+1,J-1}, u_{i+1,J}, v_{I,j-1}, v_{I,j}, v_{I,j+1})$

and:

$$f_{i,J}^{x} = \tfrac{1}{2}\delta y_{j,j+1}\rho C_T(a_k)\left[\frac{u_k^{r,\perp}(\gamma)}{1-a_k}\right]^2\cos(\gamma_k), \qquad f_{I,j}^{y} = \tfrac{1}{2}\delta y_{J-1,J}\rho C_T(a_k)\left[\frac{U_k^{r,\perp}(\gamma)}{1-a_k}\right]^2\sin(\gamma_k),$$
$$u_k^r = \sqrt{u_{i,J}^2 + v_{I,j}^2}\cos(\gamma_k - \phi_k)$$

# Appendix B

# Partial derivatives in Wind Farm Simulator (WFSim)

- $\frac{\partial J}{\partial \tilde{q}}$, with $J$ defined in Equation (2-9)

$$\frac{\partial J}{\partial \tilde{q}} = \begin{bmatrix} (J_2)_{q_1} & (J_3)_{q_2} & \cdots & (J_{N_p})_{q_{N_p-1}} & 0 \end{bmatrix} \in \mathbb{R}^{1 \times N_p \cdot n_q} \tag{B-1}$$

where $J_k$ is defined in Equation (4-22). This leads to:

$$(J_{k+1})_{q_k} = \frac{\partial J_{k+1}}{\partial q_k} = \begin{bmatrix} 0 & \cdots & 0 & \frac{\partial J_{k+1}}{\partial u_{n=1,r=1}} & \cdots & \frac{\partial J_{k+1}}{\partial u_{n=1,r=m}} & 0 & \cdots \\ & & 0 & \frac{\partial J_{k+1}}{\partial u_{n=N,r=m}} & \cdots & \frac{\partial J_{k+1}}{\partial u_{n=N,r=m}} & 0 & \cdots & 0 \end{bmatrix} \; in \; \mathbb{R}^{1 \times n_q} \tag{B-2}$$

with:

$$\frac{\partial J_{k+1}}{\partial u_{n=n_i,r=r_i}} = -\frac{6c_J}{m} \cdot \frac{u_{n=n_i,r=r_i} \cdot \left( \sum_{r=1}^{m} \sqrt{u_{n,r}^2 + v_{n,r}^2} \right)}{\sqrt{u_{n=n_i,r=r_i}^2 + v_{n=n_i,r=r_i}^2}} \tag{B-3}$$

- $\frac{\partial C}{\partial \tilde{q}}$, with $C$ as:

$$C(\tilde{q}, \tilde{w}) = \begin{bmatrix} C_1(q_0, q_1, w_1) \\ C_2(q_1, q_2, w_2) \\ \vdots \\ C_{N_p}(q_{N_p-1}, q_{N_p}, w_{N_p}) \end{bmatrix} = \; \in \; \mathbb{R}^{N_p \cdot n_q \times 1} \tag{B-4}$$

with $C_k$ defined in Equation (4-20). This results in:

$$\frac{\partial C}{\partial \tilde{q}} = \begin{bmatrix} (C_1)_{q_1} & & & & \\ (C_2)_{q_1} & (C_2)_{q_2} & & & \\ & \ddots & \ddots & & \\ & & (C_{N_p-1})_{q_{N_p-2}} & (C_{N_p-1})_{q_{N_p-1}} & \\ & & & (C_{N_p})_{q_{N_p-1}} & (C_{N_p})_{q_{N_p}} \end{bmatrix} \in \mathbb{R}^{N_p \cdot n_q \times N_p \cdot n_q} \tag{B-5}$$

with:

$$(C_k)_{q_k} = A \in \mathbb{R}^{N_p \times N_p}, \qquad (C_k)_{q_{k-1}} = -\bar{A}(q_k, q_{k-1}, w_k) \in \mathbb{R}^{N_p \times N_p} \tag{B-6}$$

with $A$ defined in Equation (2-18) and $\bar{A}(q_k, q_{k-1}, w_k)$ in Equation (2-20).

- Using Equations (B-1) and (B-4) we can obtain $\Lambda \in \mathbb{R}^{N_p \cdot n_q \times 1}$:

$$\left(\frac{\partial C}{\partial \tilde{q}}\right)^T \Lambda = -\left(\frac{\partial J}{\partial \tilde{q}}\right)^T \tag{B-7}$$

$$\begin{bmatrix} ((C_1)_{q_1})^T & ((C_2)_{q_1})^T & & & \\ & ((C_2)_{q_2})^T & \ddots & & \\ & & \ddots & \left((C_{N_p-1})_{q_{N_p-2}}\right)^T & \\ & & & \left((C_{N_p-1})_{q_{N_p-1}}\right)^T & \left((C_{N_p})_{q_{N_p-1}}\right)^T \\ & & & & \left((C_{N_p})_{q_{N_p}}\right)^T \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_{N_p-1} \\ \lambda_{N_p} \end{bmatrix} = - \begin{bmatrix} (J_2)_{q_1} \\ (J_3)_{q_2} \\ \vdots \\ (J_{N_p})_{q_{N_p-1}} \\ 0 \end{bmatrix} \tag{B-8}$$

- $\frac{\partial J}{\partial \tilde{w}}$:

$$\frac{\partial J}{\partial \tilde{w}} = \begin{bmatrix} (J_1)_{w_1} & (J_2)_{w_2} & \cdots & (J_{N_p})_{w_{N_p}} \end{bmatrix} \in \mathbb{R}^{1 \times N_p \cdot N} \tag{B-9}$$

with:

$$(J_k)_{w_k} = \frac{\partial J}{\partial w_1} = \begin{bmatrix} -c_J \left(U^{r,\perp}_{n=1,k}\right)^3 & -c_J \left(U^{r,\perp}_{n=2,k}\right)^3 & \cdots & -c_J \left(U^{r,\perp}_{n=N_p,k}\right)^3 \end{bmatrix} \in \mathbb{R}^{1 \times N} \tag{B-10}$$

- $\frac{\partial C}{\partial \tilde{w}}$:

$$\frac{\partial C}{\partial \tilde{w}} = \begin{bmatrix} (C_1)_{w_1} & & & \\ & (C_2)_{w_2} & & \\ & & \ddots & \\ & & & (C_{N_p})_{q_{N_p}} \end{bmatrix} \in \mathbb{R}^{N_p \cdot n_q \times N_p \cdot N} \tag{B-11}$$

with:

$$(C_k)_{w_k} = \frac{\partial C_k}{\partial w_k} = -\bar{B}(q_{k-1}, w_k) \in \mathbb{R}^{n_q \times n_w} \tag{B-12}$$

where $\bar{B}(q_{k-1}, w_k)$ is defined in Equation (2-20).

- Using Equations (B-8), (B-9) and (B-11), we obtain the gradient $\bigtriangledown_{\tilde{w}} J$:

$$\bigtriangledown_{\tilde{w}} J = \frac{\partial J}{\partial \tilde{w}} + \Lambda^T \frac{\partial C}{\partial \tilde{w}} \in \mathbb{R}^{1 \times N_p \cdot N} \tag{B-13}$$

# Appendix C

# Additional Simulation Results

In this appendix, the simulation results that were mentioned but not shown in Chapter 6 will be given.

As mentioned in Chapter 6, three different wind farms will be evaluated in this thesis. In this section, the three different wind farms will be discussed shortly. The size of the grid an of the cells in the grid are shown in Table 6-3.

## C-1 Wind Farm Topologies

In this section, the wind farm topologies for the 3-, 4- and 6-turbine wind farms will be given. Table C-1 shows the dimensions of the 3-turbine wind farm.

**Table C-1:** Size of the grid for the 3-turbine wind farm.

| | | |
|---|---|---|
| Number of turbines | $N$ | 3 |
| Rotor diameter [m] | $D$ | 90 |
| Length in x-direction [m] | $L_x$ | 3000 |
| Length in y-direction [m] | $L_y$ | 1250 |
| Locations of rotor center ($x$) [m] | $R_x$ | $\begin{pmatrix} 500 & 500+7D & 500+14D \end{pmatrix}$ |
| Locations of rotor center ($y$) [m] | $R_y$ | $\begin{pmatrix} 0.5L_y & 0.5L_y & 0.5L_y \end{pmatrix}$ |
| Number of grid points ($x$) | $N_x$ | 50 |
| Number of grid points ($y$) | $N_y$ | 25 |

Figure C-1 shows the resulting grid including the wind turbine locations.

The second wind farm is the 4-turbine wind farm. The dimensions of this wind farm are given in Table C-2 and a schematic representation is given in Figure C-2.

The final wind farm considered is a 6-turbine wind farm consisting of two rows of 3 turbines. Table C-3 gives the dimenions of this wind farm. Figure C-3 gives a schematic representation of this wind farm.

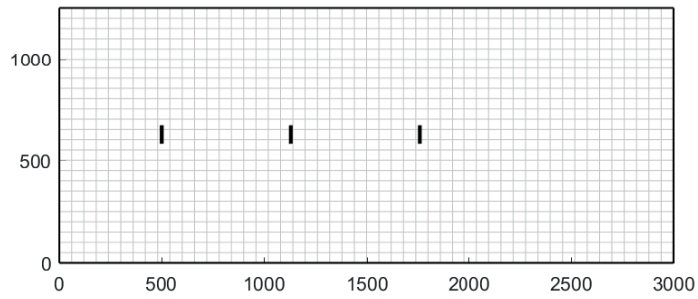



**Figure C-1:** Representation of the grid with 3 turbines in WFSim.

**Table C-2:** Size of the grid for the 4-turbine wind farm.

| | | |
|---|---|---|
| Number of turbines | $N$ | 4 |
| Rotor diameter [m] | $D$ | 90 |
| Length in x-direction [m] | $L_x$ | 3000 |
| Length in y-direction [m] | $L_y$ | 1250 |
| Locations of rotor center ($x$) [m] | $R_x$ | $\begin{pmatrix} 500 & 500+7D & 500+14D & 500+21D \end{pmatrix}$ |
| Locations of rotor center ($y$) [m] | $R_y$ | $\begin{pmatrix} 0.5L_y & 0.5L_y & 0.5L_y & 0.5L_y \end{pmatrix}$ |
| Number of grid points ($x$) | $N_x$ | 50 |
| Number of grid points ($y$) | $N_y$ | 25 |

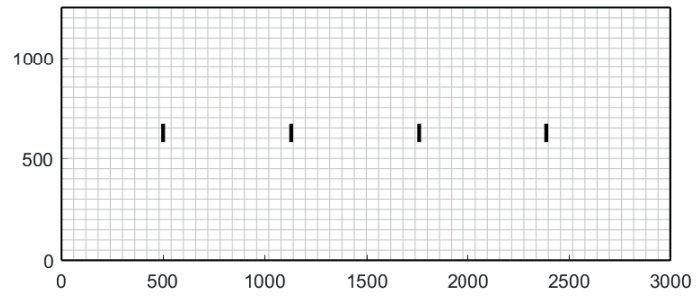



**Figure C-2:** Representation of the grid with 4 turbines in WFSim.

**Table C-3:** Size of the grid for the 4-turbine wind farm.

| | | |
|---|---|---|
| Number of turbines | $N$ | 6 |
| Rotor diameter [m] | $D$ | 90 |
| Length in x-direction [m] | $L_x$ | 3000 |
| Length in y-direction [m] | $L_y$ | 1250 |
| Locations of rotor center ($x$) [m] | $R_x$ | $\begin{pmatrix} 500 & 500 & 500+7D & 500+7D & 500+14D & 500+14D \end{pmatrix}$ |
| Locations of rotor center ($y$) [m] | $R_y$ | $3 \times \begin{pmatrix} 0.5L_y - 3.5D & 0.5L_y + 3.5D \end{pmatrix}$ |
| Number of grid points ($x$) | $N_x$ | 50 |
| Number of grid points ($y$) | $N_y$ | 25 |

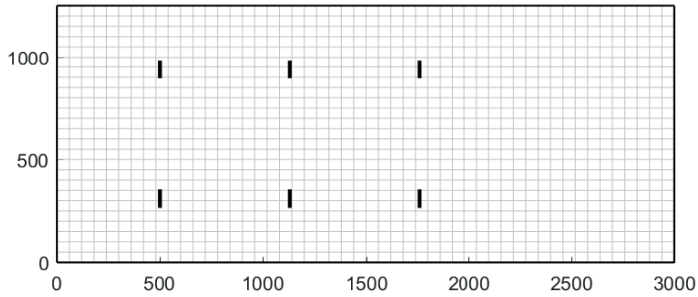



**Figure C-3:** Representation of the grid with 6 turbines in WFSim.

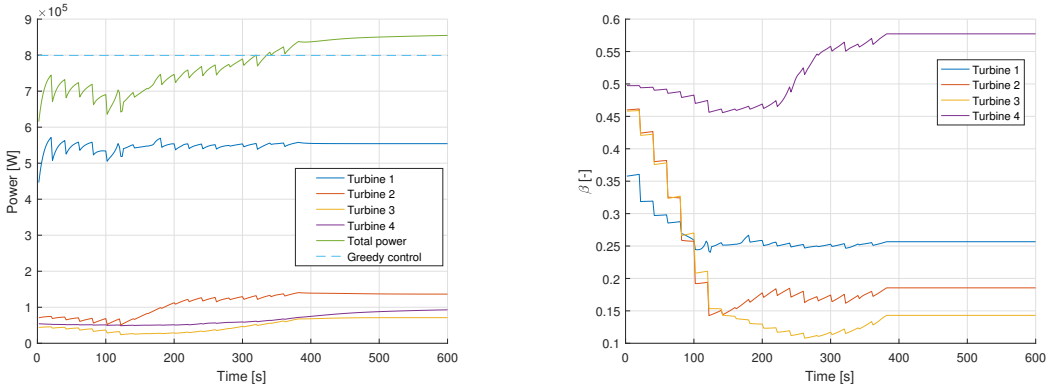
### C-1-1 Case 1: Constant Atmospheric Conditions

First of all, the results for case 1 will be presented. The atmospheric conditions and control settings are given in Table C-4.

**Table C-4:** The flow and control settings used in case 1.

| | | |
|---|---|---|
| Inflow velocity (x-direction) | $u_b$ | 8m/s |
| Inflow velocity (y-direction) | $v_b$ | 0m/s |
| Initial pressure | $p_0$ | 0Pa |
| Time step size | $\Delta t$ | 2s |
| Prediction horizon | $N_p$ | 400s |
| Receding horizon | $N_u$ | 20s |
| Number of receding horizons | $N_{u,\max}$ | 30 |
| Maximum number of line searches | $i_{\max}$ | 10 |
| Threshold of the cost function | $\epsilon_p$ | 0.002 |

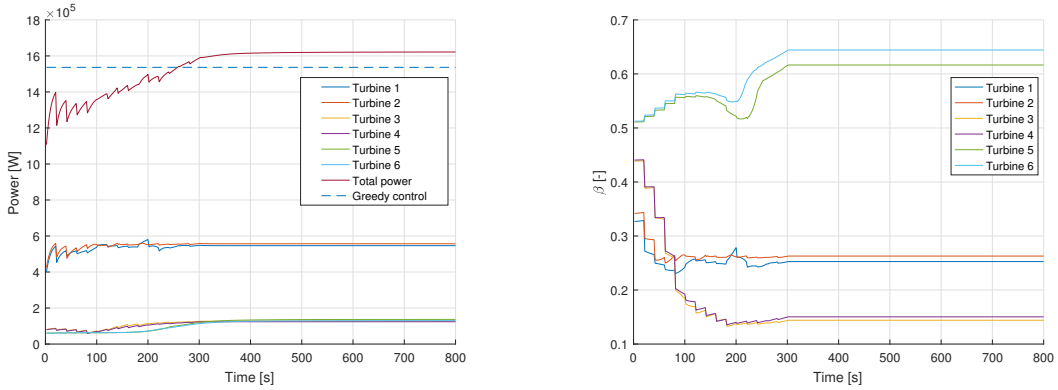
The simulation results of the 4-turbine wind farm for this case - both power production and control input - are given in Figure C-4. The power production is 7% higher compared to greedy control.




**(a)** Power **(b)** Input

**Figure C-4:** The power production (a) and corresponding input (b) of a 4-turbine WF.

The same simulation has been done for the 6-turbine wind farm. These results are given in Figure C-5. For this layout, the increase in power production compared to greedy control is 5.5%.




**(a)** Power **(b)** Input

**Figure C-5:** The power production (a) and corresponding input (b) of a 6-turbine WF.

## C-1-2 Case 2: Changing Inflow Wind Velocity

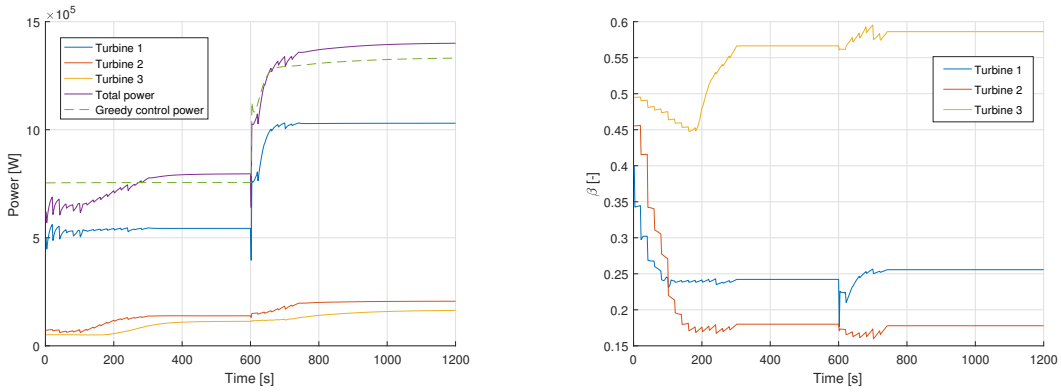
Secondly, the results for case 2 will be presented. The atmospheric conditions and control settings are given in Table C-5.

**Table C-5:** The flow and control settings used in case 2.

| | | |
|---|---|---|
| Inflow velocity (x-direction) | $u_b$ | $\begin{cases} 8\text{m/s} & \text{if } i \leq 0.5N_{u,\max} \\ 10\text{m/s} & \text{if } i > 0.5N_{u,\max} \end{cases}$ |
| Inflow velocity (y-direction) | $v_b$ | 0m/s |
| Initial pressure | $p_0$ | 0Pa |
| Time step size | $\Delta t$ | 2s |
| Prediction horizon | $N_p$ | 400s |
| Receding horizon | $N_u$ | 20s |
| number of receding horizons | $N_{u,\max}$ | 60 |
| Maximum number of line searches | $i_{\max}$ | 10 |
| Threshold of the cost function | $\epsilon_p$ | 0.002 |

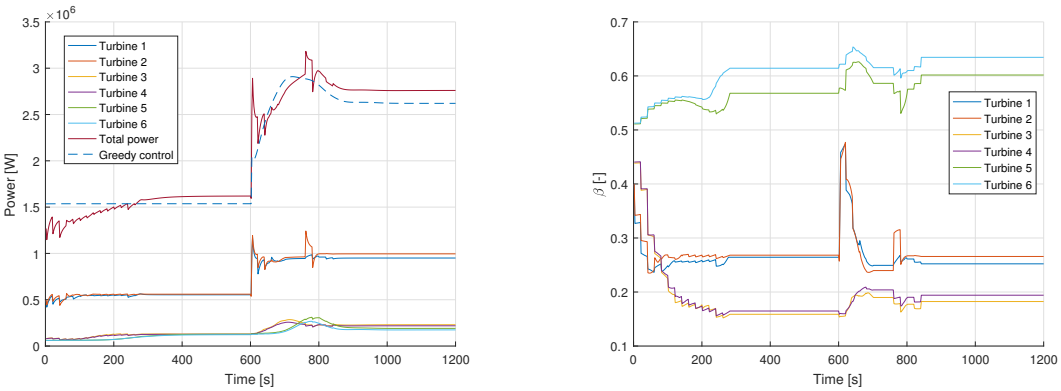
The simulation results of the 3-turbine wind farm for this case - both power production and control input - are given in Figure C-6. The power production is 6.4% higher compared to greedy control.

The same simulation has again been done for the 6-turbine wind farm. These results are given in Figure C-7. For this layout, the increase in power production compared to greedy control is 5.3%.




**(a)** Power **(b)** Input

**Figure C-6:** The power production (a) and corresponding input (b) of a 3-turbine WF.




**(a)** Power **(b)** Input

**Figure C-7:** The power production (a) and corresponding input (b) of a 6-turbine WF.

### C-1-3 Case 2: Changing Inflow Angle

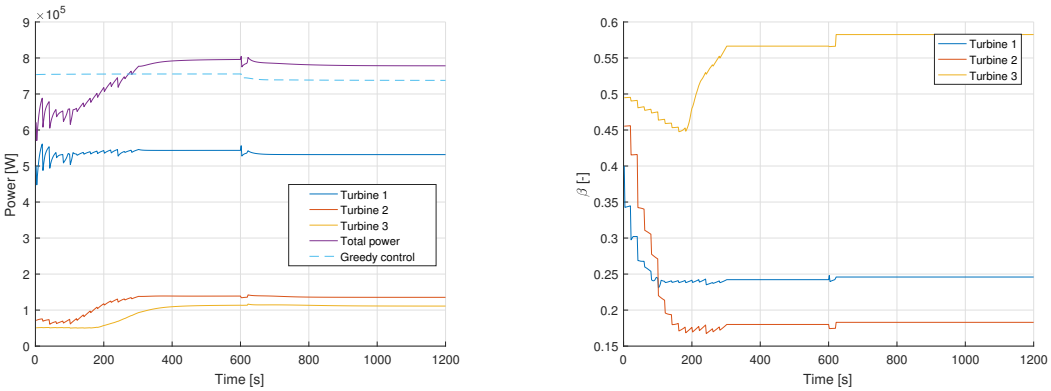
Finally, the results for case 3 will be presented. The atmospheric conditions and control settings are given in Table C-6.

The simulation results of the 3-turbine wind farm for this case - both power production and control input - are given in Figure C-6. The power production is 5.4% higher compared to greedy control.

The same simulation has again been done for the 4-turbine wind farm. These results are given in Figure 6-6. For this layout, the increase in power production compared to greedy control is 6.4%.

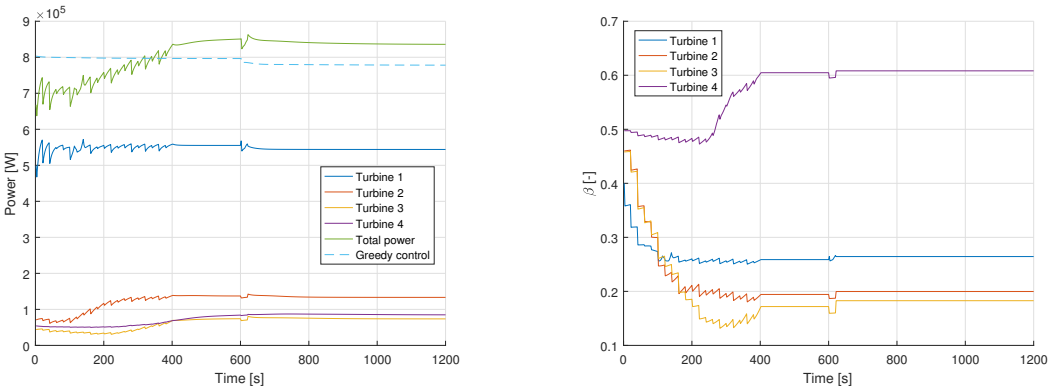
**Table C-6:** The flow and control settings used in case 3.

| | | |
|---|---|---|
| Inflow velocity (x-direction) | $u_b$ | $\begin{cases} 8\text{m/s} & \text{if } i \leq 0.5N_{u,\max} \\ 8 \cdot \cos(30°)\text{m/s} & \text{if } i > 0.5N_{u,\max} \end{cases}$ |
| Inflow velocity (y-direction) | $v_b$ | $\begin{cases} 0\text{m/s} & \text{if } i \leq 0.5N_{u,\max} \\ 8 \cdot \sin(30°)\text{m/s} & \text{if } i > 0.5N_{u,\max} \end{cases}$ |
| Initial pressure | $p_0$ | 0Pa |
| Time step size | $\Delta t$ | 2s |
| Prediction horizon | $N_p$ | 400s |
| Receding horizon | $N_u$ | 20s |
| Number of receding horizons | $N_{u,\max}$ | 60 |
| Maximum number of line searches | $i_{\max}$ | 10 |
| Threshold of the cost function | $\epsilon_p$ | 0.002 |




**(a)** Power **(b)** Input

**Figure C-8:** The power production (a) and corresponding input (b) of a 3-turbine WF.




**(a)** Power **(b)** Input

**Figure C-9:** The power production (a) and corresponding input (b) of a 4-turbine WF.

# Bibliography


[1] B. Obama, "The irreversible momentum of clean energy," *Science*.

[2] W. Obergassel, C. Arens, L. Hermwille, N. Kreibich, F. Mersmann, H. E. Ott, and H. Wang-Helmreich, "Phoenix from the ashesâĂŤan analysis of the paris agreement to the united nations framework convention on climate change," *Wuppertal Institute for Climate, Environment and Energy*, vol. 1, pp. 1–54, 2016.

[3] L. Y. Pao and K. E. Johnson, "A tutorial on the dynamics and control of wind turbines and wind farms," in *American Control Conference, 2009. ACC'09.*, pp. 2076–2089, IEEE, 2009.

[4] T. Knudsen, T. Bak, and M. Svenstrup, "Survey of wind farm control, power and fatigue optimization," *Wind Energy*, vol. 18, no. 8, pp. 1333–1351, 2015.

[5] S. Boersma, B. Doekemeijer, P. Gebraad, P.M.O. Fleming, J. Annoni, A. Scholbrock, J. Frederik, and J. van Wingerden, "A tutorial on control-oriented modeling and control of wind farms," in *American Control Conference (ACC)*, 2017.

[6] J. Annoni, P. Seiler, K. Johnson, P. Fleming, and P. Gebraad, "Evaluating wake models for wind farm control," in *2014 American Control Conference*, pp. 2517–2523, IEEE, 2014.

[7] T. Göçmen, P. van der Laan, P.-E. Réthoré, A. P. Diaz, G. C. Larsen, and S. Ott, "Wind turbine wake models developed at the technical university of denmark: A review," *Renewable and Sustainable Energy Reviews*, vol. 60, pp. 752–769, 2016.

[8] B. Sanderse, S. Pijl, and B. Koren, "Review of computational fluid dynamics for wind turbine wake aerodynamics," *Wind Energy*, vol. 14, no. 7, pp. 799–819, 2011.

[9] J. Marden, S. D. Ruben, and L. Y. Pao, "A model-free approach to wind farm control using game theoretic methods," *Control Systems Technology, IEEE Transactions on*, vol. 21, no. 4, pp. 1207–1214, 2013.

[10] P. Gebraad and J. Wingerden, "Maximum power-point tracking control for wind farms," *Wind Energy*, vol. 18, no. 3, pp. 429–447, 2015.

[11] N. Jensen, "A note on wind turbine interaction," *Risoe National Laboratory, Roskilde, Denmark, Technical Report No. M-2411*, 1983.

[12] M. J. Churchfield, S. Lee, J. Michalakes, and P. J. Moriarty, "A numerical study of the effects of atmospheric and wake turbulence on wind turbine dynamics," *Journal of turbulence*, no. 13, p. N14, 2012.

[13] J. Annoni, P. M. Gebraad, A. K. Scholbrock, P. A. Fleming, and J.-W. v. Wingerden, "Analysis of axial-induction-based wind plant control using an engineering and a high-order wind plant model," *Wind Energy*, 2015.

[14] Á. Jiménez, A. Crespo, and E. Migoya, "Application of a LES technique to characterize the wake deflection of a wind turbine in yaw," *Wind energy*, vol. 13, no. 6, pp. 559–572, 2010.

[15] B. Doekemeijer, J. van Wingerden, S. Boersma, and L. Pao, "Enhanced kalman filtering for a 2d cfd ns wind farm flow model," in *Journal of Physics: Conference Series*, vol. 753, p. 052015, IOP Publishing, 2016.

[16] S. Boersma, P. Gebraad, M. Vali, B. Doekemeijer, and J. van Wingerden, "A control-oriented dynamic wind farm flow model: Wfsim," in *Journal of Physics: Conference Series*, vol. 753, p. 032005, IOP Publishing, 2016.

[17] S. Boersma, M. Vali, M. Kühn, and J.-W. van Wingerden, "Quasi linear parameter varying modeling for wind farm control using the 2D Navier-Stokes equations," in *American Control Conference (ACC), 2016*, pp. 4409–4414, American Automatic Control Council (AACC), 2016.

[18] M. Ellis, H. Durand, and P. D. Christofides, "A tutorial review of economic model predictive control methods," *Journal of Process Control*, vol. 24, no. 8, pp. 1156–1178, 2014.

[19] A. Bradley, "Pde-constrained optimization and the adjoint method," 2013.

[20] R. Roth and S. Ulbrich, "A discrete adjoint approach for the optimization of unsteady turbulent flows," *Flow, turbulence and combustion*, vol. 90, no. 4, pp. 763–783, 2013.

[21] J. Goit and J. Meyers, "Optimal control of energy extraction in wind-farm boundary layers," *Journal of Fluid Mechanics*, vol. 768, pp. 5–50, 4 2015.

[22] M. Vali, J. van Wingerden, S. Boersma, V. Petrović, and M. Kühn, "A predictive control framework for optimal energy extraction of wind farms," in *Journal of Physics: Conference Series*, vol. 753, p. 052013, IOP Publishing, 2016.

[23] A. Betz, "Theoretical limit for best utilization of wind by wind-motors," *Magazine for the Entire Turbine System*, vol. 20, pp. 307–309, 1920.

[24] P. Pratumnopharat and P. S. Leung, "Validation of various windmill brake state models used by blade element momentum calculation," *Renewable energy*, vol. 36, no. 11, pp. 3222–3227, 2011.

[25] H. K. Versteeg and W. Malalasekera, *An introduction to computational fluid dynamics: the finite volume method*. Pearson Education, 2007.

[26] S. Engell, "Feedback control for optimal process operation," *Journal of process control*, vol. 17, no. 3, pp. 203–219, 2007.

[27] T. G. Hovgaard, K. Edlund, and J. B. Jørgensen, "The potential of economic MPC for power management," in *49th IEEE Conference on Decision and Control (CDC)*, pp. 7533–7538, IEEE, 2010.

[28] M. Diehl, R. Amrit, and J. B. Rawlings, "A lyapunov function for economic optimizing model predictive control," *IEEE Transactions on Automatic Control*, vol. 56, no. 3, pp. 703–707, 2011.

[29] R. Amrit, J. B. Rawlings, and D. Angeli, "Economic optimization using model predictive control with a terminal cost," *Annual Reviews in Control*, vol. 35, no. 2, pp. 178–186, 2011.

# Glossary

## List of Acronyms

| | |
|---|---|
| **2D** | 2-dimensional |
| **ADM** | Actuator Disk Model |
| **AIC** | Axial Induction Control |
| **AIF** | Axial Induction Factor |
| **CFD** | Computational Fluid Dynamics |
| **DUT** | Delft University of Technology |
| **NS** | Navier-Stokes |
| **MPC** | Model Predictive Control |
| **PDE** | Partial Differential Equation |
| **WF** | Wind Farm |
| **WFSim** | Wind Farm Simulator |

## List of Symbols

| | |
|---|---|
| $\alpha$ | Step size in the direction of gradient $\bigtriangledown_{\tilde{w}} J$ |
| $\epsilon$ | Convergence threshold |
| $\epsilon_p$ | Convergence threshold for increasing objective $J$. |
| $\mu$ | Dynamic viscosity |
| $\phi$ | Wind direction |
| $\rho$ | Air density |

| | |
|---|---|
| $\beta$ | Scaled Axial Induction Factor (AIF), used as control input |
| $\gamma$ | Yaw angle |
| $\mathbf{f}$ | Thrust force between wind and turbine |
| $\partial_\bullet$ | The partial derivative w.r.t. $\bullet$ |
| $a$ | AIF |
| $a^\bullet_{\bullet,\bullet}$ | Coefficient in the momentum equations |
| $A_r$ | Rotor area |
| $B_c$ | Boundary condition vector |
| $c_J$ | Product of constants in $J$ |
| $C_P$ | Power coefficient |
| $D$ | Rotor diameter |
| $J$ | Objective function |
| $L_x$ | Length of the wind farm in x-direction |
| $L_y$ | Length of the wind farm in y-direction |
| $m$ | Number of cells coinciding with a rotor |
| $N$ | Number of turbines |
| $N_c$ | Control horizon |
| $n_q$ | Size of state vector $q$ |
| $n_w$ | Size of input vector $w$ |
| $N_x$ | Number of grid points in x-direction |
| $N_y$ | Number of grid points in y-direction |
| $P$ | Power production |
| $p$ | Pressure |
| $q$ | States of the system |
| $q_{atm}$ | The atmospheric conditions |
| $r$ | Reference signal |
| $S_m$ | External forces vector |
| $U^r$ | Wind velocity at turbine rotor |
| $U_\infty$ | Wind velocity far upstream |
| $w$ | Input signal |
| $x$ | Horizontal axis of the farm as seen from above |
| $y$ | Vertical axis of the farm as seen from above |
| $z$ | Output signal |
| $\triangle t$ | Sampling time |
| $\triangle V$ | Volume of one cell |
| $\mathrm{N}_s$ | Number of time steps |
| u | Wind velocity in x-direction |
| v | Wind velocity in y-direction |
| $_i$ | Cell index in x-direction |
| $_j$ | Cell index in y-direction |

| | |
|---|---|
| $k$ | Discrete time instant |
| $n$ | Turbine index |
| $r$ | Index of a cell coinciding with a rotor blade |
| $r$ | Index of a cell coinciding with a rotor |