# Optimizing Wind Farm Control Strategies to Minimize Wake Loss Effects

by

ANDREW KARL SCHOLBROCK

B.S., University of Wisconsin Madison, 2008

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

Of the requirement for the degree of

Master of Science

Department of Mechanical Engineering

2011

This thesis entitled:
Optimizing Wind Farm Control Strategies to Minimize Wake Loss Effects
written by Andrew Karl Scholbrock
has been approved for the Department of Mechanical Engineering

_____
Gary Pawlas

_____
Patrick Moriarty

_____
Lucy Pao

Date_____

The final copy of this thesis has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.

Scholbrock, Andrew Karl (M.S. Mechanical Engineering)

Optimizing Wind Farm Control Strategies to Minimize Wake Loss Effects

Thesis directed by Professor Gary E. Pawlas

Wind farms have been used on large scales to increase the amount of power produced while minimizing the interconnection costs to decrease the overall cost of energy. However placing these wind turbines in close proximity to one another has reduced the performance of wind turbines due to wake loss effects. By modeling wind turbines using actuator disk theory and modeling wakes using the PARK and Mosaic Tile wake models, two different optimization methods are used in this research to determine the ideal axial induction factor configuration in order to minimize wake loss effects to improve the overall performance of a wind farm. Lowering the axial induction factor allows for more wind to pass through the upstream wind turbines reducing the wake loss effects so that downwind turbines can produce more power. Through this research it has been shown that with the optimization of the axial induction factors a 4% to 6% increase in power can be obtained depending on the size of the wind farm, the turbine spacing, and the number of turbines controlled. Additionally comparisons are made to the wake loss effects observed at the Horns Rev wind farm as well as wind farm control experiments performed in a wind tunnel at the Energy research Center of the Netherlands. With this further study may be performed to extend the optimized axial induction factor configurations to determine optimized pitch and rotor speed control strategies.

# Acknowledgements

I would like to take this opportunity to thank all of those that have provided me with support throughout the duration of this research project in order to make it possible. First I would like to thank Professor Gary Pawlas whose mentorship as my research advisor has guided me through many of the questions that I set out to answer in this project and who has also provided me with great motivation. I would also like to thank Dr. Patrick Moriarty at the National Wind Technology Center who has proved to be an invaluable resource with all of his knowledge of the existing research that has been done with wind turbine wakes. I would also like to acknowledge the Wind Turbine Wake Virtual Laboratory funded through the POW'WOW project and the National Science Foundation as well as DONG Energy and Vattenfall for providing my with the Horns Rev wind farm data which has been very useful in making comparisons between the results from my research to actual wind farm data. Finally I would like to thank the Department of Mechanical Engineering at the University of Colorado at Boulder for providing me with MATLAB which has been a critical tool used throughout my research in order to conduct it effectively.

# CONTENTS

## Chapter

1 Introduction 2

2 Background Information 6

2.1 Wind Turbine Performance Characteristics ……..……..6

2.1.1 Actuator Disk Theory ……..……6

2.1.2 Wind Turbine Control……..……...9

2.2 Wind Turbine Wakes ……9

2.2.1 Individual Wakes ……..…...9

2.2.2 Wake Effects in Wind Farm …..……11

2.3 Wake Modeling ……...14

2.3.1 Distributed Roughness Elements ……...15

2.3.2 Kinematic Models ……...15

2.3.2 Field Models ……...18

2.3.4 RANS, LES, and DNS CFD Models ……...19

2.3.5 Commercial Software ……...20

2.3.6 Validation of Wake Models ……...21

2.4 Optimization Methods ……...23

2.4.1 Genetic Algorithm Method ……...24

2.4.2 Monte Carlo Method ……...26

2.4.3 Pattern Search Method ……..26

2.5 Wind Farm Optimization ……..27

2.5.1 Layout Optimization ……..28

2.5.2 Wind Tunnel Experiments of Pitch Control Optimization ……..29

3 Methods 33

3.1 Turbine and Wake Model Implementation ……..33

3.1.1 Turbine Model Implementation ……..34

3.1.2 Wind Farm Layout Implementations ……..35

3.1.3 Wake Model Implementation ……..36

3.1.4 Generalization for Arbitrary Wind Farms ……..38

3.1.5 Wake and Turbine Model Validation ……..40

3.2 Optimization Method Implementation ……..41

3.2.1 Genetic Algorithm Implementation..……..42

3.2.2 Pattern Search Implementation ……..44

3.2.3 Optimization Method Comparison ……..44

4 Results 46

4.1 Linear Wind Farm Results ……..46

4.1.1 Optimization Control Comparisons ……..47

4.1.2 Wind Farm Size and Number of Rows Controlled ……..52

4.2 Wind Farm Array Results ……..54

4.2.1 Square Lattice with Rotating Wind Direction ……..54

4.2.2 Hexagonal Lattice Wind Farm ……..57

4.2.3 Horns Rev Wind Farm Optimization ……...58

4.2.4 Optimized Layout and Wind Farm Control ……...59

5 Conclusions 60

References 65

## Appendix

A Program Flow Diagrams ……...68

A.1 Wake Model Validation ……...68

A.2 Turbine/Wake Coupled Model Validation ……...69

A.3 Wind Farm Control Optimization Method ……...70

B Code Implementation ……...71

B.1 `testFindInputVelocities.m` ……...71

B.2 `twoTurbineOptimalAxial.m` ……...73

B.3 `optimalWindFarmComputation.m` ……...76

B.4 `findWindFarmFitness.m` ……...80

B.5 `findTotalPower.m` ……...81

B.6 `findIndividualPower.m` ……...81

B.7 `findInputVelocities.m` ……...82

B.8 `findPARKInputVelocities.m` ……...82

B.9 `findMosaicTileInputVelocities.m` ……...86

# TABLES

**Table**

2.1 Data Line/Wind Direction Correlation ……...12

2.2 Pitch Control Configurations ...……...30

3.1 Surface Roughness Values ……..……...37

3.2 Genetic Algorithm Options ……..…...43

4.1 Wake Model Performance Comparisons ……..……...49

4.2 ECN Analytical Model Comparison ……..……..51

4.3 Horns Rev Optimization Performance Improvements ……..……...58

# FIGURES

**Figure**


2.1 Stream tube Analysis ……..……..……7

2.2 Actuator Disk Performance Curves ……..……..9

2.3 Wake Profile Expansion ……..……10

2.4 Wake Interaction Types ……..……11

2.5 Layout of the Horns Rev, Nysted, and Middelgrunden Wind Farms ……..……12

2.6 Horns Rev Velocity Deficit ……..……..13

2.7 Nysted Velocity and Power Deficit Comparison ……..……13

2.8 Turbulence Intensity Effect on Efficiency ……..……14

2.9 PARK Model Schematic ……..……16

2.10 Wake Model Survey ……..……21

2.11 Wake Cross Section ……..……22

2.12 Example Optimization Problem ……..……23

2.13 ECN Heat and Flux Performance Results……..……31

2.14 ECN Axial Induction Factor Simulation ……..……31

3.1 Linear Wind Farm Turbine Layout ……..……35

3.2 Square Lattice Wind Farm Layout ……..…36

3.3 Hexagonal Lattice Wind Farm Layout ……..……36

3.4 Mosetti Optimized Wind Farm Layout ……..……36

3.5 Horns Rev Wind Farm Layout ……..……36

3.6 Vector Geometry for Arbitrary Wind Farms ……..……39

3.7 Wake Model Validation ……..……40

3.8 Performance Model Validation ……..……41

3.9 Optimization Method Comparison ……..……44

4.1 Wake Model Comparison for a 3 Turbine Row ……..……47

4.2 Wake Model Comparison for a 5 Turbine Row ……..……47

4.3 Wake Model Comparison for a 10 Turbine Row ……..……47

4.4 Wake Model Comparison of Velocity Deficits ……..……...48

4.5 Power Performance Comparison ……..……49

4.6 Cumulative Power Performance Comparison ……..……50

4.7 Comparison to ECN Analytical Model ……...50

4.8 Optimization Comparison to ECN's Wind Tunnel Experiments ……..…51

4.9 Optimized Performance Increase vs. Turbine Spacing ……..……51

4.10 Optimization Increase vs. Wind Farm Size ……..……52

4.11 Optimization Increase vs. Number of Turbines Controlled ……..……53

4.12 Optimized Square Lattice Results for a Rotating Wind Direction …..……56

4.14 Hexagonal Lattice Wind Farm Layout ……..……57

4.15 Performance Increase for a Hexagonal Lattice Wind Farm ……..……57

4.16 Horns Rev Wind Farm Layout ……..……58

4.17 Horns Rev Optimization for a 221.4° Wind Direction ……..……58

4.18 Horns Rev Optimization for a 270° Wind Direction ……..……58

4.19 Horns Rev Optimization for a 312.2° Wind Direction ……..……58

## Nomenclature

| | |
|---|---|
| $a$ | Axial Induction Factor |
| $\alpha$ | Wake Spreading Angle |
| $A$ | Turbine Area |
| $\beta$ | Downstream Turbine Angle |
| $C_P$ | Power Coefficient |
| $C_T$ | Thrust Coefficient |
| $d$, $x$ | Downstream Distance |
| $D$ | Turbine Diameter |
| $\Gamma$ | Wake Adjustment Coefficient |
| $h$ | Turbine Hub Height |
| $\kappa$ | Wake Spreading Coefficient |
| $\rho$ | Density |
| $p$ | Pressure |
| $P$ | Power |
| $\Psi$ | Added Wake Expansion |
| $R$ | Turbine Radius |
| $\theta$ | Wind Direction |
| $u$, $v$ | Wind Velocity |
| $Vel\_def$ | Velocity Deficit |
| $z_0$ | Surface Roughness |

# Chapter 1

## Introduction

Increasing the efficiency of wind farms is necessary to reduce the dependence on other more traditional unsustainable energy methods. The demand for energy around the globe has never been higher and it continues to rise each year. In order to keep up with the demand many countries rely heavily on fuels such as coal, oil, gas, uranium, etc. in order to produce power. These methods to produce energy are not sustainable as they produce harmful refuse in the forms of greenhouse gas emissions, and nuclear waste. The increased levels of greenhouse gases in the atmosphere have in turn led to global warming producing rapid unpredictable climate changes affecting all forms of life [1]. Additionally much care must be taken with nuclear waste in order to properly dispose of it as it is very hazardous to human health.

Many countries are looking to reduce their dependence on these waste producing energy methods by increasing their use of renewable energy. For instance, the United States set out in 2006 to increase its energy efficiency and develop a more diversified energy portfolio [2]. This led to a national goal to provide 20% of the U.S. electricity from wind energy alone by 2030 [2]. In 2010 10% of the total energy being used in the U.S. came from renewable energy sources most of which came from hydroelectric power [3]. However, it is feasible for the U.S. to meet this goal and reduce its greenhouse gas emissions with currently existing technologies [4].

One of these current energy technologies that could be improved to be more efficient is wind energy. Like any electrical power generation system, wind turbines need a source of energy. This energy source is from the momentum of the air in the wind. In order to capture

more energy out of the wind, many turbines may be built in order to take as much air momentum and transfer it to electrical power. Placing these turbines in close proximity to one another is necessary in order to reduce the electrical cable connections as well as the amount of service roads that need to be built to support the operations and maintenance of the wind turbines. This in turn increases the cost effectiveness of wind farms. However since wind turbines are extracting energy out of the wind they create a momentum deficit downwind of the turbine known as a wake. If another turbine is located inside this wake it will produce less power than if it was isolated by itself. Hence, placing many wind turbines in close proximity to one another results in performance decreases due to wake loss effects.

One way to improve the efficiency of wind farms is to minimize the losses that occur from wind turbines that are caught in the wakes of other wind turbines. One method to reduce wake losses is to optimize the layout of the wind farm. This is done by analyzing the topography of the wind farm site as well as prevailing wind directions and determining where the best wind turbines sites would be in order to minimize wake effects. One caveat with this method is that it needs to be done before the wind farm is built and cannot be changed afterwards.

Another method to minimize the impact of wakes on wind farms is to optimize the control strategy of the wind farm. Traditional wind farms operate by having each individual wind turbine operate at its own optimum performance. This causes a large wake to form downwind of the turbine which reduces the performance of other wind turbines. Instead, it has been suggested that if the upwind turbine operated below its optimum performance it would allow more wind to pass through and downwind turbines can then produce more power [5]. This control strategy would reduce the power produced by the upwind turbine, but increase the power produced from other wind turbines, and increase overall performance of the wind farm. This

method could be implemented on a new or existing wind farm by adding an integrated control system. Additionally it could be actively adjusted during the operation of the wind farm if the wind were to change directions.

The objective of this research looks to optimize the operational control strategies of wind farms by adjusting the performance of each wind turbine in order to minimize the wake loss effects that occur in wind farms. Different methods are used to model the wake losses and comparisons are made between these models and existing wind farm data. Comparisons are also made on how the wake models affect the optimized control strategies of the wind farms. Additionally different optimization methods are used to determine the effects that they may have on the operational strategies. Several wind farm configurations are used to illustrate different features of the results, and a discussion on how the configurations impact the optimal control strategies is also provided.

Background information on the details of how the performance of wind farms can be modeled and optimized will be discussed in Chapter 2. Approaches on how to model and control wind turbines will be considered here. Furthermore details will be given discussing different studies that have been done in developing methods for modeling wind turbine wakes and comparing these models to data collected from existing wind farms. Comparisons between different optimization methods and how they have been used in optimizing wind farms will also be explored here. In addition to this, previous work about wind farm optimization will also be surveyed.

Chapter 3 will talk about methods that were implemented for this research project in order to simulate the performance of wind farms. Discussions will be given detailing the decisions that were made throughout the research project that affected how the methods were

used. Comparisons are made with existing wind farm data and show that the wake models are in good agreement. Details will also be given on how the wake models are applied to different optimization methods in order to determine the ideal operational strategy for a wind farm with an arbitrary layout and arbitrary wind characteristics. Comparisons between these different optimization methods are shown here to not have an impact on the resulting control strategy.

Results for optimized wind farm control strategies are presented in Chapter 4. The optimized control strategies show a 4% to 6% increase in power performance when compared to traditional control strategies. Results are also given to show how the control strategy should change with changes in wind direction. Furthermore the results show that the amount of increased performance depends on the size and turbine spacing of the wind farm. Additional results are given for different two dimensional wind farm arrays, as well as existing wind farm layouts.

Conclusions made from the results as well as possibilities of future work are posed in Chapter 5. With the optimization strategies discussed here, an increase in wind farm performance can be achieved. Additional work that could be incorporated into this research includes extending the control strategies to optimize the pitch of the wind turbines as well as take into account other phenomenon that occurs in actual wind farms such as terrain, atmospheric stability, and turbulence with more complex wake models.

# Chapter 2

# Background Information

Accurate modeling of wakes produced by wind turbines is critical in determining the overall performance of wind farms. Robust optimization methods are also crucial in order to efficiently search for a global optimization of all of the parameters to be determined. This chapter will discuss the important details necessary for wake modeling and global optimization that will be required for this research project. Furthermore, wind turbine performance characteristics and control strategies will also be discussed as these will serve as the variables to be optimized in order to increase the performance of the entire wind farm. Previous work concerning layout and pitch control optimization of wind farms to minimize wake loss effects will also be surveyed here.

## 2.1 Wind Turbine Performance Characteristics

### 2.1.1 Actuator Disk Theory

Many different methods exist to determine how the wind interacts with the wind turbine. Direct simulation of the wind turbine blades requires a very refined mesh which in turn is very computationally costly. Another model to capture the performance characteristics of a wind turbine that is widely used is actuator disk theory developed by Froude [6]. Instead of modeling each rotating turbine blade, actuator disk theory alternatively treats the rotor area as a semi-permeable circular disk with a pressure drop. Actuator disk theory makes the following assumptions [7]:

1. The flow is incompressible
2. The pressure far upstream and downstream from the turbine is equal to the ambient static pressure
3. The rotor is composed of an infinite number of blades
4. The thrust is uniform over the entire rotor area
5. The wind speed immediately before and after the actuator disk is the same

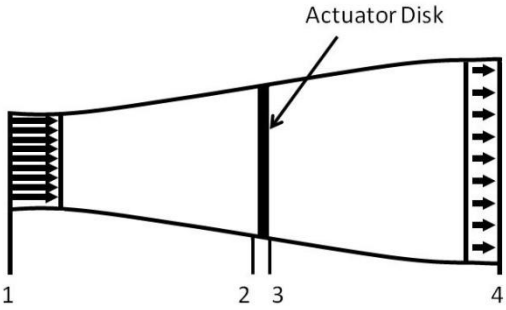



**Figure 2.1:** Stream tube analysis used in Actuator Disk Theory.

A stream tube analysis is then done for a volume of air that moves through the rotor disc as shown in Figure 2.1. Bernoulli's equation can then be applied for a streamline from 1 to 2 and from 3 to 4:

$$\text{Eqn. 2.1} \qquad p_1 + \frac{1}{2}\rho u_1^2 = p_2 + \frac{1}{2}\rho u_2^2$$

$$\text{Eqn. 2.2} \qquad p_3 + \frac{1}{2}\rho u_3^2 = p_4 + \frac{1}{2}\rho u_4^2$$

The pressure drop across the disk ($p_2 - p_3$) is solved from equations 2.1 and 2.2 using $p_1 = p_4$ and $u_2 = u_3$ from assumptions 2 and 5 respectively. The thrust then evaluated as the product of the pressure drop ($p_2 - p_3$) through the rotor disk multiplied by the rotor area ($A$). Defining the axial induction factor as the decrease in upstream velocity normalized by the upstream velocity (as seen in equation 2.3) the pressure drop can be rewritten in the form of equation 2.4:

$$\text{Eqn. 2.3} \qquad a = \frac{u_1 - u_2}{u_1}$$

$$\text{Eqn. 2.4} \qquad \frac{p_2 - p_3}{\rho} = \frac{2a}{1-a} u_2^2$$

The power is then calculated to be the thrust multiplied by the momentum of the air passing through the rotor disk. The power and thrust coefficients can then be computed:

$$\text{Eqn. 2.5} \qquad P = u_2 A(p_2 - p_3) = \frac{1}{2}\rho A u_1^3 4a(1-a)^2$$

$$\text{Eqn. 2.6} \qquad C_P = \frac{P_{turbine}}{P_{wind}} = \frac{\frac{1}{2}\rho A u_1^3 4a(1-a)^2}{\frac{1}{2}\rho A u_1^3} = 4a(1-a)^2$$

$$\text{Eqn. 2.7} \qquad C_T = \frac{\text{Thrust Force}}{\text{Dynamic Force}} = \frac{\frac{1}{2}\rho A u_1^2 4a(1-a)}{\frac{1}{2}\rho A u_1^2} = 4a(1-a)$$

A more detailed derivation of these quantities can be seen on pages 92-96 in Manwell [7]. By varying the axial induction factor the theoretical maximum power extraction occurs when $a$ = 1/3. Using equation 2.6 the maximum power coefficient is 16/27 (≈59%). This is known as the Betz limit. Figure 2.2 shows comparisons of different wind turbine performance characteristics for various axial induction factors. For axial induction factors greater than 1/2 the actuator disk theory becomes invalid [7]. Figure 2.2 shows that the peak power coefficient, $C_P$, occurs at $a$ = 1/3. Below this value, the downstream wind speed increases and the thrust coefficient decreases. Above $a$ = 1/3 the downstream wind speed decreases and the thrust coefficient increases. Hence it may be advantageous to lower the axial induction factor from 1/3 so that the wind speed deficit in the downstream wake will be less.

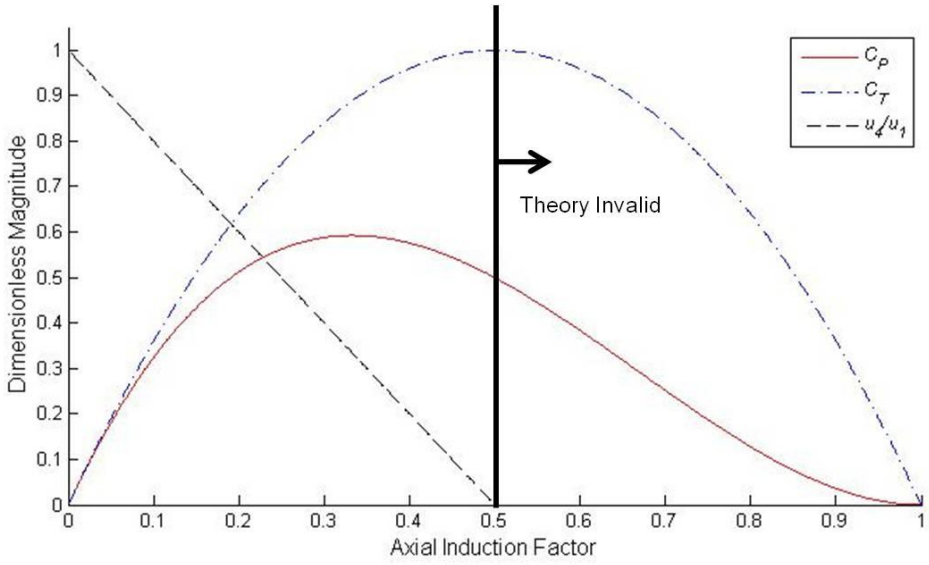



**Figure 2.2:** Comparison of the thrust and power coefficients as well as the upstream and downstream wind speeds for various axial induction factors.

### 2.1.2 Wind Turbine Control

In the operation of a wind turbine many large scale wind turbines implement two types of control schemes to optimize the performance of the wind turbine [7]. The first control scheme is to vary the rotor speed (how fast the turbine blades rotate) with the wind speed which is generally implemented for lower wind speeds. The second control scheme is to vary the pitch of the turbine blades which is generally done at higher wind speeds in order to shed excessive loads. By controlling the rotor speed and the pitch, the axial induction factor and hence the power coefficient (from equation 2.6) can be adjusted so that the wind turbine operates as close as it can to the Betz limit [7].

## 2.2 Wind Turbine Wakes

### 2.2.1 Individual Wakes

Wakes are a result of wind turbines generating energy by extracting momentum from the air and converting it into electrical energy. Since momentum is being removed from the air there is a region downstream of the wind turbine where a velocity deficit occurs. This region is known

as a wake. This deficit vanishes far downstream of the turbine due to turbulent mixing. Wake characteristics including the velocity deficit profile as well as how the wake expands and recovers downstream of the wind turbine are affected by many factors. One factor that affects the shape of a wake is the vortices that form off of the wind turbine's blade tip and root. These vortices have been shown to affect the wake expansion as well as the vortex spiral twist and the strength of the tip vortex spiral itself [8]. This affect is apparent in the near wake behind the turbine as it dissipates rather quickly. More details about other factors that affect wake properties are discussed by Vermeer in [8].

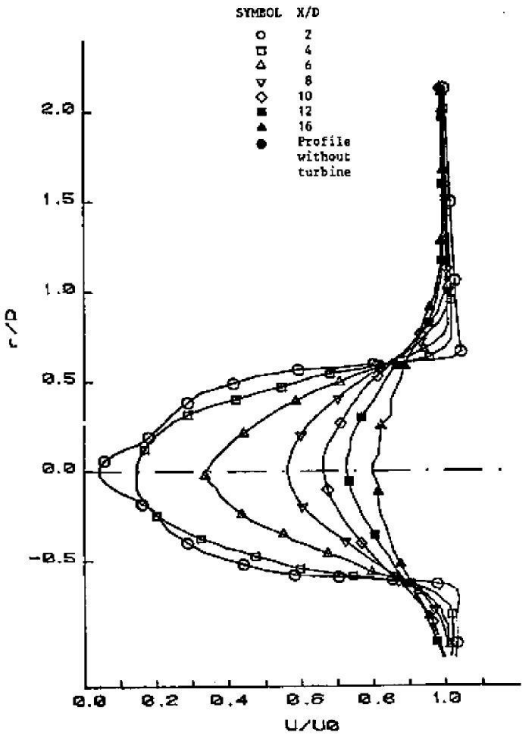



**Figure 2.3:** Wake profile at various downstream distances from wind tunnel experiments of a wind turbine [8].

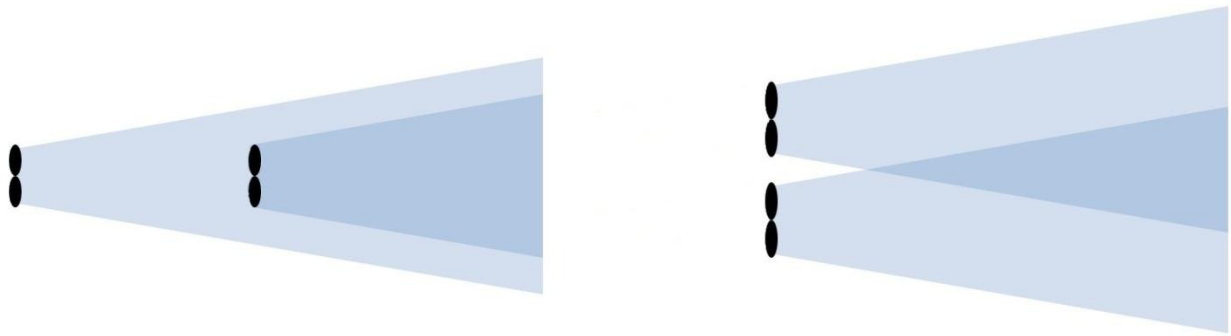
In studying the far wake, the important items to understand are the wake expansion and velocity deficit recovery since this is where other wind turbines will be located. Figure 2.3 shows the characteristics of a single wind turbine wake from experiments performed in a wind tunnel [8]. Here we see that initially there is a strong velocity deficit in the center of the wake with a steep gradient back to the reference velocity. As the wake develops further downstream the wake expands radially and the maximum velocity deficit diminishes. Note that the velocity deficit is not constant inside the wake, but instead has a smooth transition from the maximum deficit in the center to the undisturbed velocity far away. Far enough downstream the wake eventually expands to the point where it begins to interact with the surface of the ground. It has been shown

that to some extent the wake generally maintains its shape and that the wake's centerline moves up in height [9].

### 2.2.2 Wake Effects in Wind Farms

In addition to understanding some of the general features of individual wind turbine wakes, it is also important to study multiple wakes since wind farms are composed of many turbines and will produce a numerous wakes. One aspect of multiple wakes to examine is how wakes interact with one another. It is also important to investigate how multiple wakes will affect the performance of wind farms.

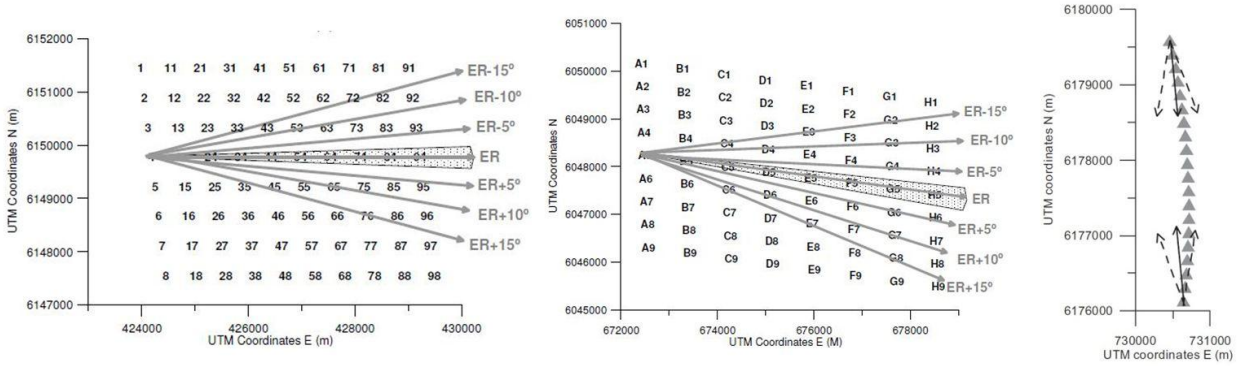
There are two main types of wake interactions that occur in wind farms and can be seen in Figure 2.4. The first type of interaction occurs when one turbine creates a wake and further downstream another turbine creates a wake inside of the first one (see part (a) of Figure 2.4). The second type of interaction occurs when two turbines create individual wakes which both expand and eventually intersect further downstream (see part (b) of Figure 2.4). While few experimental studies have been done on wake interactions it is important not to neglect this phenomenon when modeling wakes. Further description will be discussed later on how different wake models account for wake interactions.



**(a)** **(b)**

**Figure 2.4:** Illustration of the two main types of wake interactions that occur in wind farms. Part (a) shows the interaction when one wake is inside of another and part (b) shows the interaction when two wakes expand and intersect.

When one turbine is inside the wake of another turbine its performance is affected due to the wake deficit. This reduction in wind speed in turn leads to a power loss which decreases the efficiency of an entire wind farm. It has been observed that wake effects alone on average account for about a 10% loss in power [10]. When designing and operating a wind farm it is important to understand these wakes in order to minimize the impact that they will have on the performance of the wind farm.




**(a)** **(b)** **(c)**

**Figure 2.5:** Layout of the Horns Rev (a), Nysted (b), and Middelgrunden (c) wind farms. The arrows denote the wind directions for which data was collected [10-12].

**Table 2.1:** Data lines collected at the Horns Rev wind farm compared to the corresponding wind direction angles

| **Wind Direction** | 255° | 260° | 265° | 270° | 275° | 280° | 285° |
|---|---|---|---|---|---|---|---|
| **ER Data Offset Angle** | ER +15° | ER +10° | ER +5° | ER 0° | ER -5° | ER -10° | ER -15° |

Several wind farms have been studied to investigate the impact that wakes have on wind farms by collecting data of the wind speed and the power produced at each turbine. Some of the wind farms that have been studied include the Middelgrunden wind farm [10], the Nysted wind farm [11], and the Horns Rev wind farm [12] all of which are located offshore. Figure 2.5 shows the wind turbine layouts at each of these wind farms. When the wind direction was aligned with

one of the turbine rows a significant drop in performance was observed at each of these wind farms [10-12]. Figure 2.6 shows the velocity deficits averaged across the span wise direction at the Horns Rev wind farm. The wind directions in Figure 2.6 and data lines in Figure 2.5a are correlated in Table 2.1. Here we see that when the wind direction is 270° it is directly in line with the wind turbine rows and the largest velocity deficit occurs. As the wind direction becomes more misaligned with the wind turbine rows the affect of wakes decreases.

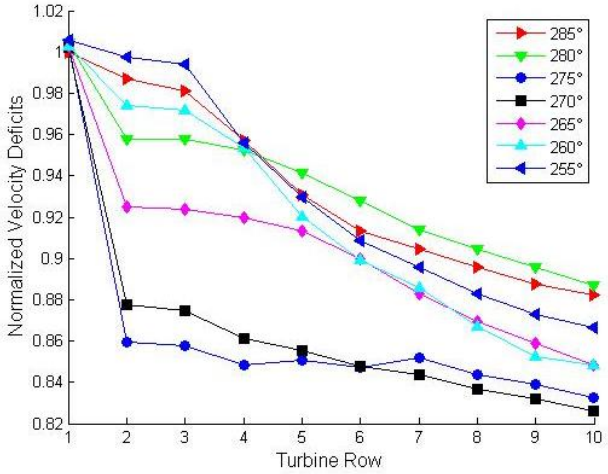



**Figure 2.6:** Velocity deficits from data collected at Horns Rev for an 8 m/s wind speed for various wind directions [13].

In addition to the velocity deficits of the wind turbines the power produced is also affected by wakes. Equation 2.5 shows that the power produced by a wind turbine is proportional to the *cube* of the velocity. In the study of the Middelgrunden wind farm the normalized power losses were compared to the normalized velocity deficits as seen in Figure 2.7 [10]. In this figure we see that the power loss observed for a north (circles) and south (x's) wind direction is even more significant of a factor than the velocity deficit observed from a southerly direction (solid diamonds).

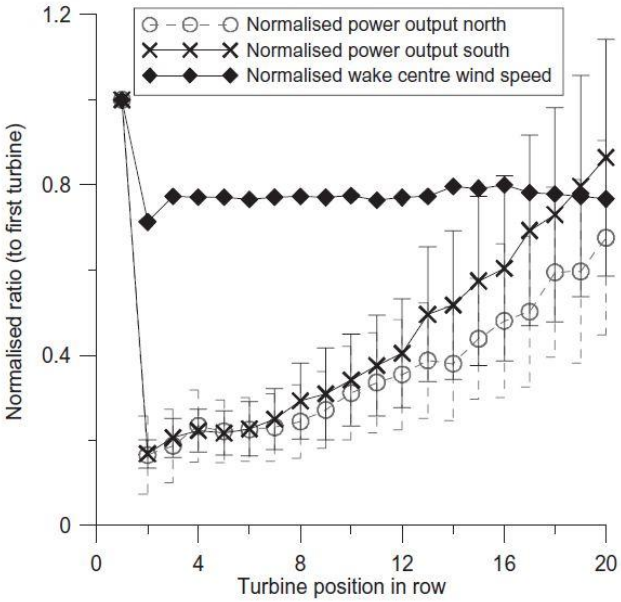



**Figure 2.7:** Power losses compared to velocity deficits for the Middelgrunden wind farm [10].

Turbulence intensity is another factor that will affect the aerodynamics of wind turbine wakes. At the Nysted wind farm a study showed that an increase in turbulence intensity an increase in the efficiency of the wind farm was observed as seen in Figure 2.8 [11]. This is because the increase in the turbulence of the wind results in the wakes dissipating quicker and reduces the power losses hence increasing the overall efficiency. This effect was said to be minimal though when compared to the overall losses caused by wind turbine wakes [11].

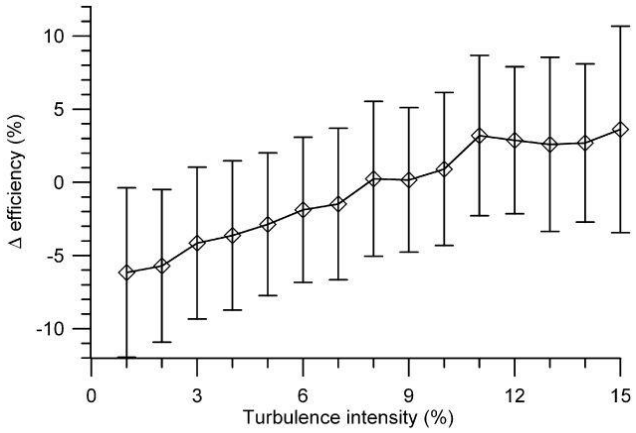



**Figure 2.8:** Difference in wind farm efficiency compared to the relative average vs. the turbulence intensity [11].

## 2.3 Wake Modeling

In addition to studying empirical wake measurements from existing wind farms and wind tunnel experiments it is equally important to try to model wind turbine wakes with computational methods so that researchers and wind farm developers can accurately simulate the performance of a wind farm without having to invest in building physical models which can be both costly and time consuming. There are many different ways in which wakes can be modeled. Some models provide very detailed results at a high computational cost and as such are limited to very small wind farm sizes. Conversely there are other models that do not provide as much detail, but have a small computational cost which allow them to be used to model much larger scale wind farms. Surveying these models is necessary in order to establish a broad knowledge of all the different aspects of modeling wind turbine wakes.

### 2.3.1 Distributed Roughness Elements

One way in which wind turbines are modeled is consider the wind turbines as a rough element that creates turbulence in the wind (much like a tree or a building). As discussed by Crespo, *et al*., this method used a logarithmic wind profile is which requires a ground roughness parameter [14]. By adding the wind turbines to the domain this roughness parameter is adjusted to include the added roughness created by the wind turbines [14]. In calculating the effect of wakes this model assumes that when the wind travels from the first row of turbines to the next row sufficient mixing occurs and the velocity deficit can be averaged from the sum of the momentum flux due to drag (power extraction) from the first row of turbines, the amount of momentum lost due to ground effects, and the amount of momentum entrained from above through mixing [14]. While this model has not been widely used it may be of some use in predicting large scale effects of an entire wind farm for the surrounding regions.

### 2.3.2 Kinematic Models

Kinematic wake models are some of the most widely used models as they are quick in performance and provide quite reasonable results when compared to wind farm data. These models are based on self-similar velocity deficit profiles and factors that affect wake spreading in order to conserve momentum in the wake behind a wind turbine [14]. Lissaman proposed a model in which the wake expands linearly as it travels downstream [15]. The cross-sectional profile of the velocity deficit wake was assumed to follow a Gaussian distribution [15]. This model was then simplified by Jensen by assuming that simply a constant velocity deficit in the wake could be used instead of a Gaussian distribution [16]. A schematic of this model can be seen in Figure 2.9. This model was then extended by Katic, *et al.* and has come to be known as

the PARK model [17]. The PARK model has been shown to be within good agreement of wind tunnel data beyond three to four turbine diameters downstream of the wind turbine [17].

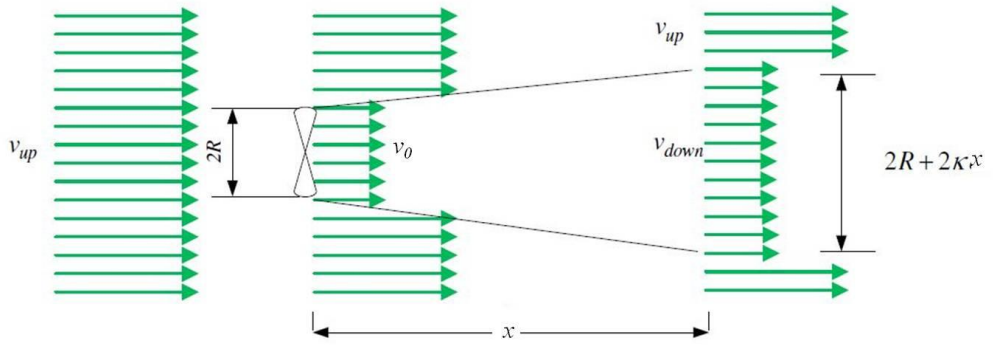



**Figure 2.9:** Diagram of the PARK wake model [18].

By assuming a conservation of momentum the PARK model equates the momentum flux through a circular area of radius $R_0$ upstream and downstream of the turbine:

$$\text{Eqn. 2.9} \qquad \pi R^2 v_0 + \pi(R_0^2 - R^2)v_{up} = \pi R_0^2 v_{down}$$

$$\text{Eqn. 2.10} \qquad R_0 = 2R + 2\kappa x$$

Here $\kappa$ is defined as the wake spreading coefficient, and $x$ is the downstream distance from the wind turbine producing the wake. Using the definition of the axial induction factor from equation 2.3, equation 2.9 can then be rearranged into the following form:

$$\text{Eqn. 2.11} \qquad v_{down} = v_{up}\left[1 - 2a\left(\frac{R}{R+\kappa x}\right)^2\right]$$

The velocity deficit can then be defined as:

$$\text{Eqn. 2.12} \qquad Vel\_def = 1 - \frac{v_{down}}{v_{up}} = 2a\left(\frac{R}{R+\kappa x}\right)^2$$

With the PARK model a velocity deficit can be calculated inside the wake as a function of the turbine diameter, $D$, downstream distance, $x$, reference velocity, $v_{up}$, and spreading coefficient, $\kappa$, as seen in equation 2.12. Frandsen went on to relate the wake spreading coefficient to the

relative surface roughness ($z_0$) by empirical means [19] which can be seen in the following equation:

$$\text{Eqn. 2.13} \qquad \kappa = \frac{0.5}{\ln\left(\frac{h}{z_0}\right)}$$

Jensen and Katic further extended the PARK model to account for multiple wakes interacting, however they do not distinguish between the two types illustrated in Figure 2.4 [16, 17]. For a location inside multiple wakes the velocity deficit is said to be the square root of the sum each individual velocity deficit squared in order to maintain a momentum balance [16, 17]:

$$\text{Eqn. 2.14} \qquad Vel\_def_{total} = \sqrt{\sum_{i=1}^{N} Vel\_def_i^{\,2}}$$

Another more recent kinematic wake model known as the Mosaic Tile Model has been developed and accounts for wake expansions and interactions differently than the PARK model [20]. Similar to the PARK model, the Mosaic Tile model assumes a self-similar velocity deficit profile and solves this by conserving momentum [21]. The velocity deficit is then calculated by the following equation [21]:

$$\text{Eqn. 2.15} \qquad Vel\_def = a\frac{A_0}{A}$$

where $a$ is the axial induction factor, $A_0$ is the rotor area, and $A$ is the area of the wake expanded downstream of the turbine with a diameter given by the following equation [9]:

$$\text{Eqn. 2.16} \qquad D = D_0\left[\max(\beta^{k/2}, \frac{\kappa x}{D_0})\right]^{1/k}\Psi$$

Here $D_0$ is the turbine diameter, $\Psi$ is a factor that allows for extra expansion for each downwind turbine passed, $k$ accounts for the rate of the wake expansion, and $\beta$ is given by the following equation [9, 20]:

$$\text{Eqn. 2.17} \qquad \beta = \frac{1 - 1/2\, a}{1 - a}$$

It has been shown that for values of $k = 3$ and $\kappa = 1.2$ and $k = 2$ and $\kappa = 0.7$ show good agreement with wind farm data [9]. The $\max(\beta^{k/2}, \frac{\kappa x}{D_0})$ term in equation 2.16 was modified by Rathmann in [20] from an earlier work by Frandsen [21] in order to account for different wake expansion characteristics between the near and far field downstream [20]. It has been noted by Frandsen that this model is still under development comparing the model parameters $k$ and $\kappa$ to wind farm data as well as including dependence on turbine operational characteristics and wake overlapping [9].

### 2.3.3 Field Models

Field models provide a more detailed description of the flow by expanding the wake solution into two dimensions [14]. Some of these models take into account additional factors that affect the flow field including atmospheric stratification, atmospheric turbulence, and Coriolis forces [14]. While kinematic models can be useful in estimating the effect of wake losses based on empirical data they do not provide any physical insight into the flow phenomena like field models do [22]. Ansilie proposed a parabolic eddy-viscosity turbulence model that assumes an axisymmetric wake flow [22]. In solving the model, boundary conditions need to be specified and numerical methods need to be implemented in order to solve for the flow field over the entire domain. Comparing the model to wind farm data showed good agreement [22].

Another field model developed by Crespo *et al.* known as UPMWAKE looks to solve the field model by immersing the wind turbines in a non-uniform flow corresponding to the surface layer of the atmospheric boundary layer [14]. This takes into account the atmospheric stability as well as the surface roughness. The equations used in this model look to conserve mass, momentum, and energy using a $k$-$\varepsilon$ method for closure of the turbulent flow equations [14].

Although these models provide more insight into the physical processes of the flow with wind turbines they require much more computation than kinematic wake models, especially with larger wind farms.

### 2.3.4 RANS, LES, and DNS CFD Models

Reynolds averaged Navier-Stokes (RANS), large eddy simulation (LES) and direct numerical simulation (DNS) are all different Computational Fluid Dynamics (CFD) implementations that investigate wind turbine wakes by solving the Navier-Stokes (N-S) equations on a fully three dimensional mesh. In the domain the pressure and velocity components need to be solved at each grid point. Furthermore special boundary conditions need to be specified in order solve these equations while maintaining realistic turbulence characteristics. Additionally the turbine either needs to be fully resolved (requiring a very fine mesh resolution) or modeled in some way. Stovall *et al.* modeled wind turbines by using actuator disk theory to represent the wind turbines [23]. In addition to this Stovall *et al.* also used LES methods to solve the N-S equations [23]. While these results provided excellent details into the flow characteristics of wind turbine wakes the computational cost was very high and limited to two wind turbines [23]. Another method that has been used to model the wind turbine in CFD simulations is the actuator line technique. This technique models each of the turbines blades as a line which can provide a better representation of a wind turbine when

compared to the actuator disk theory and computationally less expensive than fully resolving the wind turbine [24]. Despite the detailed wake results that this method gives it is still limited to small wind farm simulations due to its complexity [24].

### 2.3.5 Commercial Software

Many commercial software packages exist that provide a graphical user interface (GUI) for users to interact with different terrain features, place wind turbines, and model the performance of wind farms. These packages use different wake models in order to evaluate the wake effects and determine the overall performance of the wind farm.

The Wind Atlas Analysis and Application Program (WAsP) developed by Risø implements the Mosaic Tile model and uses meteorological data to characterize local wind climates [25]. It allows for simple terrains to be accounted for in its implementation, however complex terrains are not supported [25]. Another software package is WindFarmer developed by GL Garrad Hassan. This package calculates the wind characteristics externally using WAsP and uses Ainslie's eddy-viscosity model for determining wake effects [25]. Empirical expressions are used to model the wake turbulence as well as wake interactions [25]. WAKEFARM developed by the Energy research Centre of the Netherlands (ECN) uses a wake model based on UPMWAKE and uses parabolized Navier-Stokes equations to solve the flow field [25]. Other software packages also exist that use either kinematic models or LES/DNS models [25].

These software packages provide some convenience to the user by providing an interface that is easy to interact with so that the user can quickly determine the performance of a given wind farm. On the other hand they incorporate a "black box" in which the user cannot directly interact with the wake model, or the flow field solver which the user may want to do if the

software package does not account for a specific factor that the user would like to incorporate in his or her simulation (such as terrain or atmospheric stability factors).

### 2.3.6 Validation of Wake Models

It has been shown that there are many different techniques that have been used to model wind turbine wakes. Although they vary in complexity, each model needs to be compared to existing wind farm data in order to determine how well the model matches what physically occurs in a wind farm. Several studies have been done to compare different wake models with existing wind farm data, and show that most of the models capture the wakes effects fairly accurately.

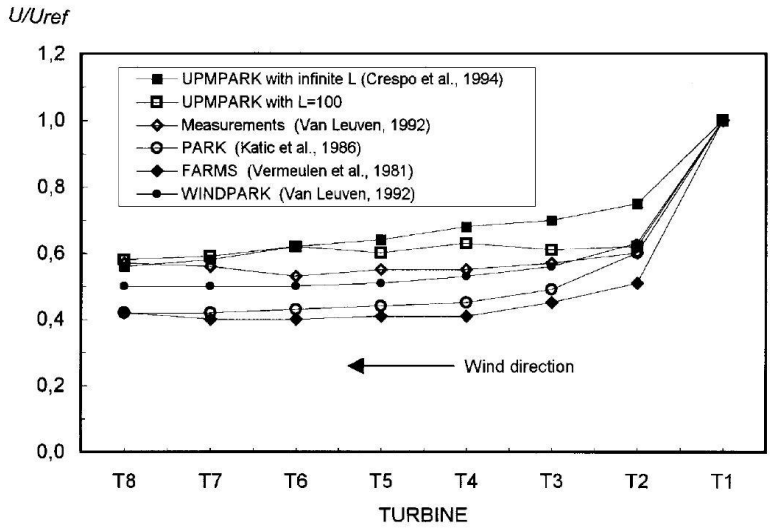



**Figure 2.10:** Survey of different wake models compared to wind farm data measurements [14].

In a survey done by Crespo several different wake models were compared to measurements at the Zeebrugge wind farm of the velocity deficits as a function of the wind turbine row as seen in Figure 2.10 [14]. It is important to note that in all of these models it is assumed that each wind turbine is operating at its own peak efficiency (with an axial induction factor close to 1/3). All of the models show that from the first turbine to the second turbine there is a significant drop in the wind speed. Beyond the second row of turbines the drop is less

significant and some models level off, where as others continue to drop slightly more. All of the models match the measured wind farm data with errors ranging from 5% to 10% for far wake calculations [14]. These results vary a little with some under predicting and some over predicting the wake effects when compared to the measured data, but all show the same general trend. Another study was done to show that the UPMWAKE field model showed good results when compared to experiments carried out at TNO [26]. Other studies have been done and show that some of the kinematic models depend strongly on the ambient turbulence intensity, while others depend heavily on the thrust coefficient [27].

In addition to validating the velocity deficits as they develop downstream some studies have been done to validate the shape of the velocity deficit profile. In Stovall's study a comparison was made between measurements taken at the Nibe wind farm to LES and RANS CFD models as well as the PARK model [23]. In Figure 2.11 it can be seen that the LES and RANS model match the measured data very well [23]. The PARK model on the other hand assumes in its implementation that there is a constant velocity deficit inside the wake. This is one advantage to more complex models since they capture more details about the shapes of the wakes. The PARK model still shows a reasonable average of the velocity deficit when compared to the measured data.

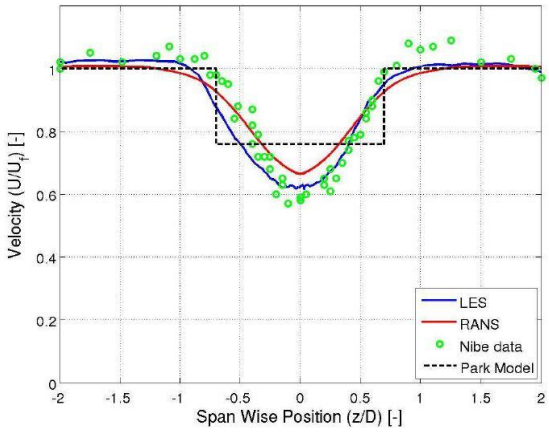



**Figure 2.11:** Cross sectional profile of a wake from different models compared to measurements from the Nibe wind farm [23].

In validating the different wake models it has been shown that the different models match existing wind farm data reasonably, with some more complex models capturing more details of the wake

characteristics than other. It has been noted that some work needs to be done to understand the causes of why some of the models under or over predict the wake loss effects [25].

## 2.4 Optimization Methods

Finding the ideal solutions for complex systems involving many different equations with a multitude of different variables can be a very computationally costly task. The goal of optimization methods is to efficiently find the global maximum or minimum of a function by using different strategies. Considering that this research project looks to find an ideal operational strategy for arbitrary wind farms it is important to compare different optimization methods since they will play a critical role in determining the optimum operational values.

Most optimization methods can be categorized as one of two types: heuristic methods and stochastic methods. Heuristic methods take advantage of patterns in a problem to develop some kind of technique that will speed up the process in finding the optimal solution [28]. Some of these methods include calculus based methods such as gradient hill climbing methods, and the extended pattern search method. Conversely stochastic methods are non-deterministic and rely on random elements to find an optimal solution [28]. Some of these methods include evolutionary algorithms such as the genetic algorithm, the Monte Carlo method, and swarm algorithms.

Heuristic and stochastic methods utilize different concepts in order to find the optimum solution of a given problem. Figure 2.12 shows an illustration of a problem with two variables that has some functional value. The goal of the optimization method is to find the optimum

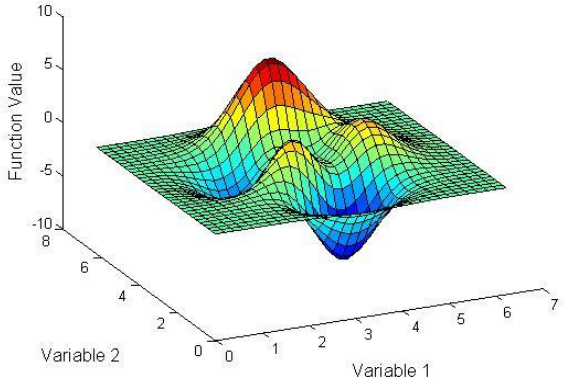



**Figure 2.12:** An example of a problem with two variables that need to be optimized.

variable values that result in the maximum functional value. Heuristic methods will utilize some kind of pattern in order to find the optimum [28]. For instance, the hill climbing technique will look at the gradient of the function wherever it starts and move in the direction of greatest increase until it reaches a peak where the gradient decreases all around that point [28]. While this method provides an efficient direct approach to optimizing the problem it can lead to a result that is the local optimum and not globally the best solution if it happens to climb the wrong peak.

On the other hand stochastic methods will optimize the problem by surveying the function at a set of random points [28]. It will then determine which points are better and from that use some method to choose the next set of points to evaluate until an optimum solution is found [28]. By having a random element in these methods they are very robust and do not have the tendency to converge to local optimums like heuristic methods do. However if the method to choose the next set of data points is not well thought out then this process can become inefficient and time consuming. Many different optimization methods exist, however for the purposes of this research project three methods were considered and are discussed in more detail in the following sub-sections.

### 2.4.1 Genetic Algorithm Method

The genetic algorithm is a stochastic method that tries to mimic natural evolutionary processes. This method has become very popular since the work of Holland [29] and now is widely used in many different types of optimization problems. Below is a list of some of the definitions that the genetic algorithm uses in its implementation [30]:

*Genetic Traits/Genes*: The variables that the genetic algorithm is trying to optimize.

*Population*: A collection of individuals with a given set of traits.

*Generation*: An iteration of the genetic algorithm process.

*Fitness Function*: A function that evaluates how well an individual satisfies the solution to the problem.

*Selection*: The process that the genetic algorithm uses to choose which individuals will get to reproduce and which ones will not.

*Reproduction*: The process of creating a new population generation using either a mutation or crossover operation.

*Mutation*: A reproduction operation in which an individual's traits are either partially or wholly chosen by a random process.

*Crossover*: A reproduction operation in which an individual's traits chosen by a recombination of traits from the previous generation.

The genetic algorithm utilizes the following general process in order to determine what combination of traits will give the best solution to the problem [30]:

1. Create an initial population of individuals
2. Evaluate the fitness of each individual of the current generation using the fitness function
3. Select which individuals will be used in the crossover operation of the reproduction process
4. Perform the reproduction process with a combination of crossover and mutation operations to create a new population generation
5. Repeat steps 2 through 4 on the next population generation until all of the individuals have converged to one another within some user defined tolerance

Each of these steps can be performed in many different ways and can be adjusted during the optimization process in order to speed it up. For example the user may know something about what the optimized solution should look like and create an initial population of individuals based on this knowledge [30]. Another example would be to vary the ratio between mutation and crossover operations in order to find a good balance between finding a globally optimized solution and how controlled vs. random the optimization process is [30]. More details about all of the factors that can be adjusted in the genetic algorithm can be found in [30].

### 2.4.2 Monte Carlo Method

The Monte Carlo method is another stochastic process that gets its name from the Monte Carlo Casino since the method involves chance. Like the genetic algorithm it relies on evaluating some fitness function at each step in the iteration [31]. However the method in which it arrives at its optimized solution is much different. The basic method of the Monte Carlo method is as follows [31]:

1. Start at a random point in the variable domain.
2. Evaluate the fitness of the current location.
3. Select a new location in the variable domain at random.
4. Evaluate the fitness at the new random point.
5. Move to the new random point if one of the two following conditions are met:
    a. The new location has a better fitness value.
    b. The difference in fitness values between the new and current location is less than some tolerance criteria set by the user.
6. Repeat steps 3 through 5 until a new location cannot be found after a number of iterations determined by the user.

This method has been very popular due to its rapid implementation process. It also moves through iterations quicker than the genetic algorithm since there is much less data bookkeeping and function evaluations to be done. However the algorithm requires the user to define some tolerance criteria on whether or not the algorithm should move to the new location which is not always obvious for complex problems involving many different variables [31]. Additionally the method relies more on randomness which may lead to slow performance.

### 2.4.3 Pattern Search Method

The Pattern Search method is a heuristic method that is similar to the hill climbing technique discussed earlier, but also incorporates methods to follow ridges in order to avoid

converging to a local optimization [28]. Like the other methods discussed a fitness function is necessary to progress to an optimum solution. The pattern search method makes adjustments during its optimization based on the history of successful moves [32]. A move is considered successful if it moves towards the optimization goal [32]. The method looks to optimize the variables by using the following method [32]:

1. Start at an initial location in the domain of the variables
2. Evaluate the fitness of the current location
3. Move in a direction along one of the variables
4. Evaluate the new location's fitness as a success or a failure
5. Determine the size of the step the algorithm should take next based on whether or not the current move was successful or not.
6. Based on a pattern of successes and failures of recent moves determine which variable the algorithm should move along next.

This method can be very useful in finding the global optimization of a problem without relying on random processes. However it has been shown that the method converges to the global optimum only for certain classes of problems [33]. Outside of these problems the solution is not always guaranteed to converge to a global solution as it may only find a locally optimized value [33].

## 2.5 Wind Farm Optimization

By combining computational wake models with different types of optimization methods there have been many studies done to determine the optimization of wind farms on a given terrain taking several different factors into account (namely wind turbine wakes). Different studies have looked to optimize wind farms in different ways. Some studies look to optimize the

wind turbine locations for a given plot of land in a wind farm. Other studies look into optimizing the pitch control of individual turbines to reduce wake loss effects.

### 2.5.1 Layout Optimization

Although this research project looks to optimize the operational methods of a wind farm it is still prudent to survey the work that has been done in wind farm layout optimization since both will utilize methods for wake modeling and optimization. The simple solution to layout optimization would be to place each turbine as far away from one another as possible in order to minimize wake loss effects on wind farm performance. Most layout optimization studies have used objective functions, such as the cost of energy in order to make this problem more interesting. Hence these studies have to take into account factors other than wake effects such as cable interconnection and service road costs when evaluating the cost of energy for a given wind farm. Many different studies have been done using different cost models, optimization techniques, and wake models. The next few paragraphs describe some of the studies that use the same wake models and optimization methods to determine the optimal wind turbine locations as the methods that were employed in this research project.

One of the first layout optimization studies investigated by Mosetti has since provided a standard for other studies to compare with. This study looked to maximize the energy captured by a wind farm while minimizing the installation cost [34]. The study modeled wind turbine wakes by implementing the PARK model, and optimization was performed using the genetic algorithm [34]. A wake spreading coefficient based on the surface roughness as proposed by Frandsen was used [19, 34]. Three different cases were then examined for different wind conditions: constant wind speed and direction, constant wind speed and variable direction, and

variable wind speed and direction. The results of this study showed that by optimizing the layout of the wind farm the cost of energy could be lowered to improve the efficiency of the wind farm.

Mosetti's study was later revisited by Grady where different genetic algorithm parameters were used in the optimization method [35]. In addition to the methods that Mosetti employed, a heuristic approach was used to find the optimal solution for the case where the wind speed and direction was held constant [35]. The study showed that while the spacing patterns were similar to that of Mosetti, adding in more turbines than what Mosetti had resulted in a greater power production to installation cost ratio [35].

Another study done by Marmidis *et al.* optimized the layout of a wind farm using the PARK model for capturing wake effects and using the Monte Carlo method for optimization [36]. The same cost model used by Mosetti was employed in this study [36]. Results showed that a lower cost of energy could be obtained when compared to Mosetti and Grady [36].

The pattern search method has also been used in wind farm layout optimization to maximize the power to cost ratio. Du Pont *et al.* performed a layout optimization study using the PARK model, the same cost model utilized by Mosetti, and a pattern search optimization algorithm [37]. Results of the study were compared to that of Mosetti, Grady, and Marmidis. The results showed a further optimized turbine layout with a higher power to cost ratio when compared to the previous studies [37]. It also should be noted that the layout patterns produced by this optimization method seemed to produce more of a regular lattice pattern than the other studies which produced more of a random layout structure.

### 2.5.2 Wind Tunnel Experiments of Pitch Control Optimization

Another method to improve the efficiency of a wind farm is to shed the power produced by a turbine allowing more wind to pass though so that subsequent turbines will produce more

power and increase the overall power produced. There have been some studies that investigate this by building a scale model wind farm in a wind tunnel and adjust the pitch on certain turbines in order to see how these changes affect the total power performance. Considering that the goal of this research project is to optimize the control scheme for a generalized wind farm, a review of some of the previous work done in this area will be surveyed here.

The Energy research Centre of the Netherlands (ECN) built a 1:400 scale model wind farm that consisted of 14 (2 turbines in the span wise direction and 7 turbines in the stream wise direction) turbines using the NACA 0012 and 0009 airfoils [38]. The goal was to test how pitching the blades would affect the overall power produced. ECN referred to this as a "Heat and Flux" operation [38]. The turbine blades were constructed such that a +0.0° pitch corresponded to the turbine operating as close as it could to the Betz limit ($a$ = 1/3) [38]. Increasing the pitch angle would result in a reduction of the axial induction factor [38]. Additionally, great care was taken to reproduce the conditions of an offshore wind farm which included matching the Reynolds number, surface roughness, and turbulence intensity values [38]. Different pitch configurations were then set up in the wind tunnel and overall power measurements were taken. These pitch configurations are listed in Table 2.2:

**Table 2.2**: Pitch Control Configurations [38]

| | Farm Layout | | |
|---|---|---|---|
| Heat and Flux Units | Turbine 1 | Turbine 2 | Turbines 3-7 |
| -1 | -2.5° | +0.0° | +0.0° |
| 0 | +0.0° | +0.0° | +0.0° |
| 1 | +2.5° | +0.0° | +0.0° |
| 2 | +2.5° | +2.5° | +0.0° |
| 3 | +5.0° | +5.0° | +0.0° |
| 4 | +7.5° | +7.5° | +0.0° |

The power measurements from each of the six experimental runs were then compared to a farm in classic operation where each turbine had a pitch angle of 0° (i.e. each turbine operating

with an axial induction factor close to 1/3). These results are illustrated in Figure 2.13. The results showed that by pitching the front two rows by +2.5°, a 2.0% increase in power could be obtained. However if the blades were pitched too much then the power produced would decrease.

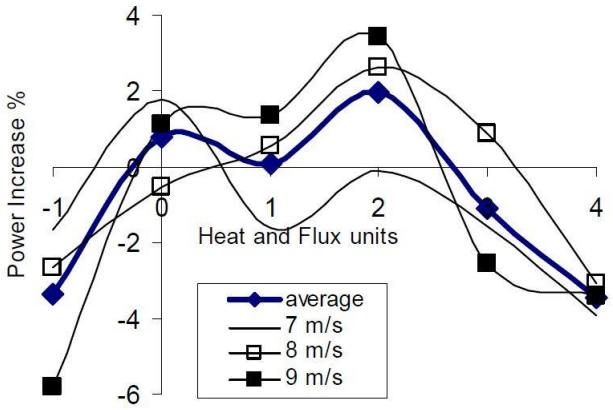



**Figure 2.13:** Experimental results of power performance of a scale model wind farm using ECN's "Heat and Flux" operational method compared to a classic wind farm operation [38].

In addition to the scale model built by the ECN a simulation was run for a simple two turbine wind farm where one wind turbine was fully inside the wake of the other [5]. The turbines were modeled in this simulation using actuator disk theory [5]. This simulation investigated how varying the axial induction factor of the front turbine would affect the overall power performance while keeping the back turbine's axial induction factor fixed at 1/3 [5]. Figure 2.14 shows the results of this simulation [5]. We see that the front turbine (P1) produces the most power when the axial induction factor is 1/3 which is consistent with actuator disk theory. However the back turbine (P2) produces very little power. The peak performance of the two wind turbines occur when the front turbine is operated at an axial induction factor of 0.2 and the back turbine at a factor of 1/3. Comparing this power to the power produced when both turbines operated at an axial induction factor of 1/3 showed that a 4.1% increase in power could

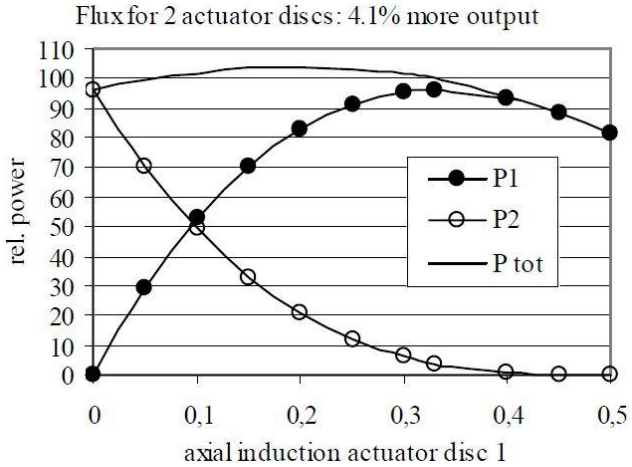



**Figure 2.14:** Power produced for a simple 2 turbine wind farm done by varying the axial induction factor of the front wind turbine [5].

be achieved [5]. Furthermore by reducing the axial induction factor of the front turbine the dynamic loads from the thrust of the wind was reduced in the front turbine [5]. This extra benefit can reduce the fatigue experienced in wind farms and improve their longevity. With these results in mind it was later proposed that a generalized control strategy would produce the optimum performance results for a row of $n$ wind turbines aligned in the wind direction [39]. This proposal said that the axial induction factor for the first turbine should equal $1/(2n+1)$, the second $1/(2n-1)$, and the following turbines $1/(2n-3)$, $1/(2n-5)$, …, 1/7, 1/5, and the final turbine equal to 1/3 [39]. Details about whether or not this would apply to arbitrary wind farm layouts and how wake interactions were accounted for in this analytical solution were not given.

While the empirical results presented by the ECN show that by pitching the blades of the first two rows increase the total power of the wind farm, it would be very time consuming to empirically explore every possible wind farm pitch control configuration in order to determine the best optimal control strategy. Hence this research project looks to extend ECN's research by computationally exploring the performance of different configurations in order to vastly expand the variable domain in hopes of finding a more optimal control strategy.

# Chapter 3

# Methods

As previously stated the goal of this research is to optimize the control strategy for a generalized wind farm in order to minimize wake loss effects. In order to accomplish this goal the first task necessary in this project is to implement a model to represent individual wind turbines as well as different methods to model the wakes that they will create. The second task is to develop a robust algorithm so that the wake model could be applied to an arbitrary wind farm layout. The wake models then need to be validated by comparing them to existing wind farm data. The third task is to implement different optimization methods in order to establish a means to optimize the wind farm control strategy. This chapter will discuss the methods that were used throughout the research project as well as the reasons behind what choices were made that affected how these methods would be implemented.

The first item that should be noted in the methods of this research is that the workspace environment used throughout this research was done through MATLAB. MATLAB was chosen at it contains many methods already programmed into it that can be utilized for this project. This includes several optimization methods as well as an extensive suite of post processing utilities. Other compiled environments (such as C++/FORTRAN) may provide better performance and could be an area of future work.

## 3.1 Turbine and Wake Model Implementation

Sections 2.1 through 2.3 illustrated that there are many different wake models and turbine models with varying degrees of complexity. The goal of this research is to look at large wind

farms and how wake losses can be avoided. With this goal in mind it is not necessary to resolve many of the details of the fluid dynamic properties of the wind if sufficient performance results could be obtained from simpler wake models. Similarly since the analysis of the results will be focused toward the operation of an entire wind farm and not the structural loading of individual turbines a simpler turbine model could be used to avoid unnecessary computations that would be required to determine specific turbine details.

### 3.1.1 Turbine Model Implementation

As discussed earlier there are several different ways in which a wind turbine can be modeled. This research project implements the actuator disk theory as described in section 2.1.1. With this model the wind turbines are represented as an actuator disk and do not have to evaluate details with individual blades. Using actuator disk theory the performance of an individual turbine is then calculated using equation 2.5. In this research project it is assumed that the air density would be constant throughout the domain with a value of 1.225 kg/m$^3$. It is also assumed that each turbine in the wind farm is identical to one another with the same performance characteristics, hub height, and rotor diameter. These values remained constant for a given simulation; however they could be adjusted to match the wind turbines used in previous work for consistency. The default wind turbine parameters that were used matched the turbines of the Horns Rev wind farm with a hub height of 70 meters and a turbine diameter of 80 meters. With these assumptions in place the remaining variables in equation 2.5 are $u$ and $a$. The velocity, $u$, serves as the input variable to be determined by the wake model, and the axial induction factor, $a$, is the variable being optimized. The undisturbed reference wind speeds used throughout these experiments were chosen to be low values (in the 7 to 10 m/s range) such that all the wind turbines would be operating in region II where the power produced was lower than the turbines'

rated power and also where the turbines are not pitching their blades to shed power, but instead trying to optimize their performance.

Using the axial induction factors as the main optimization variable was decided to be sufficient in order to investigate the general wind farm control patterns to minimize wake losses. Although using pitch as the variable to optimize would provide a more direct result that could be integrated into existing wind farms it would require additional calculations with a more complex wind turbine model during the optimization process that would slow down the performance. However this could be an area of further investigation to extend the research of this project.

### 3.1.2 Wind Farm Layout Implementations

Using the actuator disk theory to represent individual wind turbines, the next step in modeling the wind farm is to create a structure to represent all of the turbines in the wind farm and what their operation parameter is. This is done by using a matrix to store all necessary information. The matrix is *N* by 3 in size where *N* is the number of turbines in the wind farm. Each turbine in the wind farm is then represented by three elements: the first two elements are the *x* and *y* locations of each turbine, and the third element is the axial induction factor for a given turbine. Figures 3.1 through 3.5 show the layouts that have been created for different simulations throughout the research:

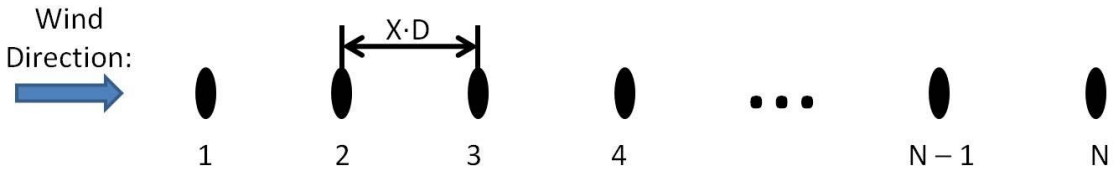



**Figure 3.1:** Turbine layout for the linear wind farm where *N* is the number of wind turbines, *D* is the turbine diameter, and *X* is an integer value used to determine the amount of spacing in between each wind turbine.

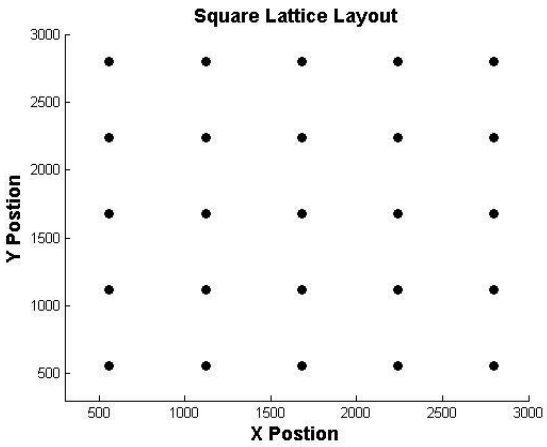



**Figure 3.2:** Turbine layout for the square lattice five by five wind farm.

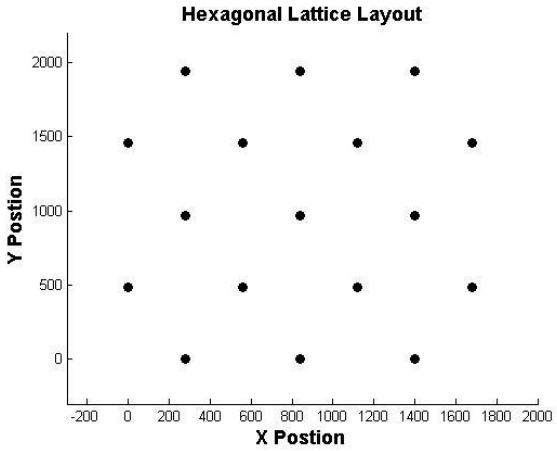



**Figure 3.3:** Turbine layout for the hexagonal lattice wind farm.

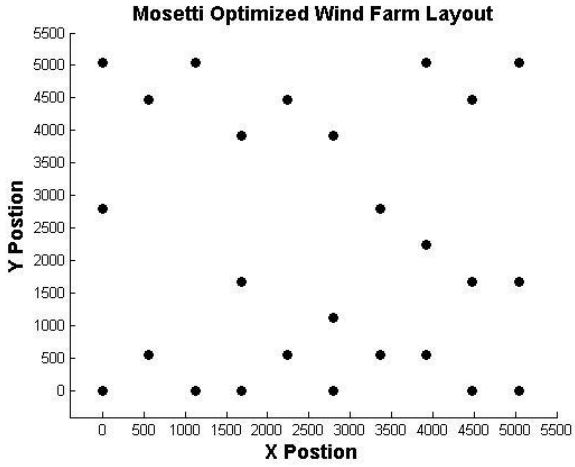



**Figure 3.4:** Turbine layout from Mosetti's optimized wind farm layout. Here the wind was blowing constantly from the north [33].

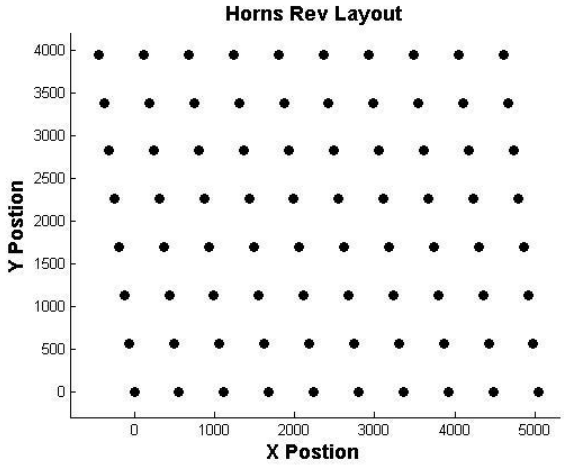



**Figure 3.5:** Turbine layout of the Horns Rev wind farm [11].

The wind direction is defined using the convention of where the wind is coming from throughout this research project. Additionally the wind direction uses a compass/wind rose style format where 0/360° is north, 90° is east, 180° is south, and 270° is west.

### 3.1.3 Wake Model Implementation

As mentioned before a simpler less complex wake model would be sufficient for this research project since specific flow details would not be necessary in evaluating the overall performance of a wind farm. With this in mind two different kinematic wake models were chosen to be implemented for this research. The PARK model was chosen because it is widely

used in a wide variety of previous work concerning wake studies. In addition to this the Mosaic Tile method was also implemented as it has been developed fairly recently and provides a different approach for handling wake interactions.

In implementing the PARK model the equations derived by Jensen (discussed in section 2.3.2) [16] are used to determine the velocity deficit at a given location inside the wake downstream of a wind turbine. Equation 2.12 has been modified in the following way:

Eqn. 3.1 $$Vel\_def = \Gamma \cdot 2a\left(\frac{R}{R+\kappa d}\right)^2$$

The Γ value has been added to the equation so that the model could be adjusted to match empirical wind farm data. A value of Γ = 0.4 shows good agreement with the model and wind farm data. Since each turbine in the wind farm is identical, the rotor radius is a constant value. The wake spreading constant is assumed to be the same value everywhere in the wind farm. This value is derived from the surface roughness using equation 2.13 [19]. Surface roughness values have been determined using values from Hau as seen in Table 3.1 [40]. The two remaining variables *a*, the axial

**Table 3.1:** Surface roughness values for various types of terrain [40].

| $z_0$ [m] | Types of terrain surfaces | Roughness class |
|---|---|---|
| 1.00 | City | |
| | Forest | |
| 0.50 | Suburbs | 3 |
| 0.30 | Built-up terrain | |
| 0.20 | Many trees and/or bushes | |
| 0.10 | Agricultural terrain with a closed appearance | 2 |
| 0.05 | Agricultural terrain with an open appearance | |
| 0.03 | Agricultural terrain with very few buildings, trees, etc. | 1 |
| | Airports with buildings and trees | |
| 0.01 | Airports, runway | |
| | Meadow | |
| $5 \cdot 10^{-3}$ | Bare earth (smooth) | |
| $10^{-3}$ | Snow surfaces (smooth growth) | |
| $3 \cdot 10^{-4}$ | Sand surfaces (smooth) | 0 |
| $10^{-4}$ | Water surfaces (lakes, fjords and the sea) | |

induction factor of the upstream turbine, and *d*, the distance between the point of interest and the upstream turbine, are the input variables that would need to be provided for each wake computation. The distance *d* is the distance between the two turbines projected onto the wind direction and can be computed using the following equation [18]:

$$\text{Eqn. 3.2} \qquad d_{ij} = \left|(x_j - x_i)\cos\theta + (y_j - y_i)\sin\theta\right|$$

Here *i* refers to the index of the upstream turbine producing the wake, and *j* refers to the index of the downstream turbine. Multiple wake interactions are all be considered in the same way regardless of the types of wake interactions. Equation 2.14 has been used to evaluate multiple velocity deficits.

Implementing the Mosaic Tile method has been done using equations 2.15, 2.16, and 2.17. The values for $k$ and $\alpha$ have been determined based on the number of turbines producing a wake at the point of interest. If only one turbine is ahead of the point of interest then a value of $k = 3$ and $\alpha = 1.2$ are used. Otherwise a value of $k = 2$ and $\alpha = 0.7$ are used. These values have been determined to model wake loss effects with good agreement with wind farm data by Frandsen [9]. The $\Psi$ term in equation 2.16 has been modified for this research project by having it equal to a constant value $\Gamma$ times the number of turbines contributing to the wake at the point of interest. A value of $\Gamma = 0.8$ has been determined to be a good value in order to match wind farm data.

### 3.1.4 Generalization for Arbitrary Wind Farms

With an arbitrary wind farm layout it is necessary to determine which wind turbines are affected by the wakes produced from other wind turbines. It is assumed that the undisturbed wind has the same speed and direction everywhere in the wind farm domain. It is also assumed that the wakes produced by the wind turbines expand linearly behind a given turbine. When

considering if a turbine is inside the wake of another turbine it is either considered to be fully inside or outside of the wake. With these assumptions a dependency matrix can then be created to list the turbines wakes that affect a given turbine.

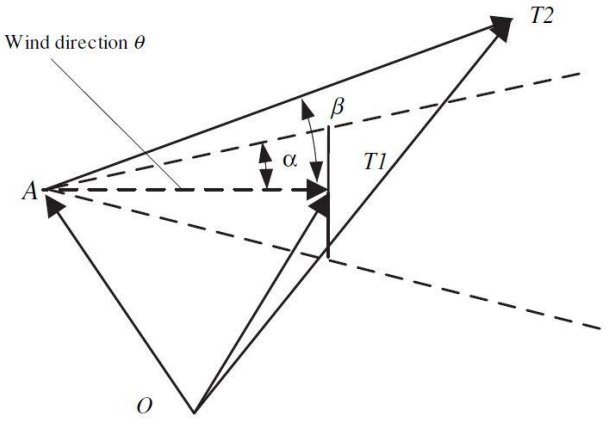



**Figure 3.6:** Vector geometry used to determine if one turbine is inside the wake of another turbine [18].

Using the vector geometry in Figure 3.6 with an arbitrary coordinate system with an origin located at *O* the angle $\beta$ can be defined as the angle between the wind direction vector originating at location *A* and ending at the upstream turbine *T1* (vector $\overrightarrow{AT1}$) and the vector originating at location *A* and ending at the downstream turbine *T2* (vector $\overrightarrow{AT2}$). If the angle $\beta$ is less than the wake spreading angle ($\alpha$) then turbine T2 is considered inside the wake of turbine T1. In order to compute the $\beta$ angle the following equation can be used [18]:

Eqn. 3.3
$$\beta_{ij} = \cos^{-1}\left\{\frac{(x_j - x_i)\cos\theta + (y_j - y_i)\sin\theta + R/\kappa}{\sqrt{(x_j - x_i + \frac{R}{\kappa}\cos\theta)^2 + (y_j - y_i + \frac{R}{\kappa}\sin\theta)^2}}\right\}$$

Here *i* refers to the index of the upstream turbine producing the wake, and *j* refers to the index of the downstream turbine. A more detailed derivation of equation 3.3 is given in [18]. With equation 3.3 a list of turbines that affect a given wind turbine can be created and stored in a dependency matrix. From this list the total velocity deficit for a given turbine can then be computed by the wake interaction method for a given wake model. A flow diagram for computing the velocity deficits for an arbitrary array can be seen in Appendix A.1.

### 3.1.5 Wake and Turbine Model Validation

Once the wake models have been implemented the next step is to validate them. This is done by comparing the wake model results to existing wind farm data to ensure that the models would match the behavior that has been empirically observed. The data that is used for this comparison came from the Horns Rev offshore wind farm located off the west coast of Denmark. The wind farm consists of 80 identical wind turbines each with a diameter of 80 meters and a hub height of 70 meters. The layout for the wind farm can be seen in Figure 3.5. The east/west turbine spacing is 7 diameters apart. In comparing the PARK model and Mosaic Tile model to the Horns Rev wind farm data Figure 3.7 shows that both models are in good agreement with the physical data with the PARK model matching the data slightly better.

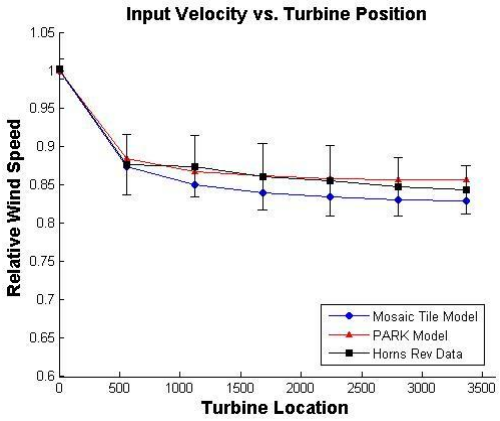



**Figure 3.7:** Comparison of the PARK model and the Mosaic Tile model to data taken from the Horns Rev wind farm.

With the wake models validated the next step is to combine the turbine model with the wake model. In order to do this an algorithm has been constructed to find the input wind velocity for each individual turbine based on the combination of wakes from the upstream turbines. With this algorithm the wind speed is then used with the wind turbines axial induction factor to determine the power produced for an individual turbine using equation 2.5. The performance of the entire wind farm is then computed as the sum of the power produced from each individual turbine. A flow diagram is provided in Appendix A.2 showing how this is done.

In order to validate this performance model a comparison has been made to ECN's simulation results seen in [5]. In order to match ECN's simulation a surface roughness value of

0 has been used. Figure 3.8 shows the performance results for two wind turbines with the back turbine's axial induction factor fixed at 1/3 and the front turbine varying from 0 to 0.5. Here we see that the front turbine maximizes its power performance when the axial induction factor is 1/3, however the back turbine produces very little power due to the large wake loss effect. The maximum power produced by both turbines occurs when the front turbine operates at $a$ = 0.2. This results in a 4.15% power increase when compared to both turbines operating at $a$ = 1/3. When comparing this figure to ECN's results (see Figure 2.14) it can be seen that both have a maximum value when the front turbine's axial induction factor is 0.2. Additionally ECN reports a 4.1% power increase which is in good agreement with the results presented here [5].

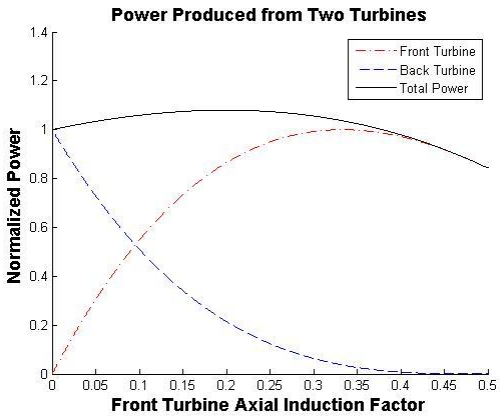



**Figure 3.8:** Power produced from two wind turbines while varying the front turbines axial induction factor and fixing the back turbine at $a$ = 1/3.

## 3.2 Optimization Method Implementation

Section 2.3 discussed some of the basics of how optimization methods work and illustrated several different optimization methods and discussed the pros and cons of the different types of algorithms that exist. Evaluating different possible objectives was also critical to ensure that the optimization method produced results that improved the overall performance of the wind farm. To ensure that the results presented by this research have not not been affected by the optimization method used, two different optimization methods were implemented and comparisons are made.

Every optimization method requires some objective to maximize or minimize through a fitness function. As mentioned before, the variables that are optimized are the axial induction

factors for each wind turbine in a given wind farm. However this still leaves the objective goal of the optimization open. Several different objectives could be used including cost of energy, wind turbine loading, total power produced, etc. Section 2.5.1 explains how many of the wind farm layout optimizations look to optimize the cost of energy by balancing the wake loss effects with the interconnection costs. For this research project it is assumed that the wind farm has already been built and the locations of each of the wind turbines are fixed. Hence optimizing the cost of energy is not necessary, but instead maximizing the power produced by the wind farm is a sufficient objective.

In choosing which optimization method to use for this project a comparison has been made between the genetic algorithm and the pattern search method. These two methods are different in that the genetic algorithm is stochastic while the pattern search method is heuristic. Since these methods have different pros and cons, it has been determined through this comparison which method would better suit the optimization for this research. In evaluating each method both computational time and global convergence are considered to determine which method to use.

### 3.2.1 Genetic Algorithm Implementation

In MATLAB, one of the items included is the global optimization toolbox. Within the toolbox one of the methods provided is the genetic algorithm [41]. This implementation of the genetic algorithm has been used for the research project. The main input to this method in MATLAB is a handle to the fitness function to be optimized. Other input variables include the number of variables to be optimized, constraints of the variables, and options that the user can specify.

It is required that the fitness function for the genetic algorithm implemented in MATLAB have a scalar output value that evaluates an individual's fitness. As mentioned before the goal of this research looks to maximize the power produced for the wind farm. Since the implementation of the genetic algorithm in MATLAB looks to find the global minimum of the fitness function, the scalar output value is the total power produced of the wind farm multiplied by negative one. This way the genetic algorithm finds the most negative value which in turn is the largest power produced. It is also required that the input to the fitness function be a vector of the variables to optimize. This vector contains all of the axial induction factors of the turbines in a given wind farm.

The constraints that have been specified for the genetic algorithm are the limits for the axial induction factors. As mentioned in section 2.1.1 the actuator disk theory being used to model the wind turbines is valid for axial induction factors in the range of $0 \leq a \leq 0.5$. This limiting factor serves as the lower and upper constraints for the genetic algorithm. Other options specified for the genetic algorithm have been determined from previous work [35]. The options that are used have been specified in Table 3.2:

**Table 3.2:** Options set for the genetic algorithm method implemented in MATLAB.

| Option: | Value: | Reason Chosen: |
|---|---|---|
| Initial Population | Varied | An educated guess on what the optimal solution should look like in order to speed up the convergence process. |
| Elite Count | 2 | Necessary for a few individuals to survive to ensure solution would converge to global optimum. |
| Tolerance | $10^{-8}$ | Convergence criteria set to stop the algorithm after the change in individuals from one generation to the next is less than this value |
| Crossover Fraction | 0.5 | Ensured an even balance between crossover regeneration and mutation regeneration to balance speed and global convergence. |
| Number of Generations | 1000 | Set to 1000 to stop the algorithm after 1000 generations if it failed to meet the tolerance criteria. |
| Population Size | 200 | Chosen so that a broad spectrum of individuals would be created to effective sample the entire solution space |

### 3.2.2 Pattern Search Implementation

Like the genetic algorithm, the pattern search method is also included in the global optimization toolbox in MATLAB. The argument list for the pattern search method is almost identical to the genetic algorithm. As such much of the same features used for the genetic algorithm could be reapplied to the pattern search method. The requirements of the fitness function for the pattern search method are identical to the genetic algorithm, so the same fitness function could be used again. Similar to the genetic algorithm, upper and lower constraints needed to be specified for the pattern search method. These constraints have been set so that the axial induction factors are constrained to be between 0 and 0.5. Other options for the pattern search method were left to the MATLAB default settings.

### 3.2.3 Optimization Method Comparison

With the two optimization method implementations, a comparison has been done to determine if either method would optimize to a different solution. Initially a comparison was done between the optimization methods and a brute force method where all possible configurations were explored for a linear wind farm with three turbines.

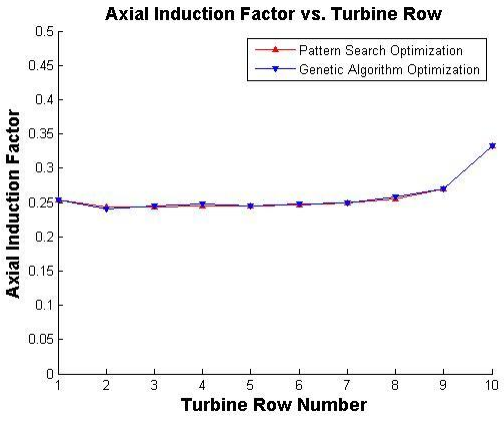



**Figure 3.9:** Comparison between the pattern search and the genetic algorithm optimization methods for a linear wind farm with 10 turbines.

The results from the optimization methods and the brute force method all showed that the axial induction factors of the first two turbines should drop to an axial induction factor around 0.25 and the last turbine optimizing to an axial induction factor of 1/3. From this the next validation step compared the optimization methods on a larger scale. Since the brute force method becomes quite computationally expensive it was not used in this comparison. Figure 3.9 shows

the optimized results for a linear wind farm with ten turbines. It can be seen that the axial induction factors for each turbine optimize to the same values regardless of which optimization method is used. They also both show the same power performance improvement of 5.62%. Hence, while the pattern search method is heuristic, it is still able to find the global optimum for the problems of this research project. Additionally this shows that the results are not affected by the optimization method used.

A comparison has also been done with different initial populations to determine if it would have an effect on the final optimized result. Three different initial populations were considered: one with all axial induction factors at zero, one with all axial induction factors at 0.5, and one with all axial induction factors at 1/3 (which is the industry strategy). In each of these cases the optimization methods converged to the same optimized solution. As such it has been determined that the initial population is not a factor that would affect the results.

In comparing the computational time it took for the genetic algorithm to optimize to the final solution with the pattern search method, it has been observed that the pattern search method would perform quicker. For a wind farm with ten turbines the genetic algorithm took approximately five minutes to run, whereas the pattern search method would take about 30 seconds. With larger wind farms the performance difference was less, however the pattern search method was still quicker. Since both methods converged to the same results, it has been decided that the pattern search method is sufficient in finding the optimized axial induction factor configuration for wind farm control strategies since it converges to the solution faster. With the optimization method implemented it was then combined with the turbine and wake models to produce a program that is able to find an optimal axial induction factor configuration for an arbitrary wind farm. A flow diagram of this program can be seen in Appendix A.3

# Chapter 4

# Results and Analysis

The results presented in this chapter are organized into two separate parts. First, results for a linear wind farm are presented followed by results for various wind farm arrays. In section 3.2.3 it has been shown that the two optimization methods produce the same results. Since these two methods produce identical outcomes, the pattern search method has solely been used for the results presented here. The linear wind farm results show that different optimization control schemes should be implemented depending on which wake model was used. Analysis of these results is discussed about where these discrepancies may be coming from. Furthermore results are given that compare the optimized wind farm performance to the traditional wind farm industry strategy. Comparisons are also made between ECN's analytical model and their empirical wind tunnel experiments [38]. In the second part of this chapter, results are presented for an array of wind farms. Using an optimization method results show that performance increases can be obtained for the Horns Rev wind farm. Other results show that performance increases can be achieved for a wind farm with an optimized layout configuration as well as a wind farm with a hexagonal lattice structure. Additionally results are presented that show how the operational strategy changes with a varying wind direction.

## 4.1 Linear Wind Farm Results

The results produced from the linear wind farm layout have been created from a row of wind turbines that are all aligned with the wind direction as illustrated in Figure 3.1. This section shows comparisons that have been made between the different wake models that have

been used as well as comparisons to the ECN results and how much the optimized performance increase varies with the wind farm size and number of wind turbines controlled.

### 4.1.1 Optimization Control Comparisons

In section 3.1.4 the PARK model and Mosaic Tile model have both been validated by comparing these two models to the Horns Rev wind farm data. Furthermore, in section 3.2.3, it has been shown that the genetic algorithm and the pattern search optimization methods agreed with each other and did not have an impact on the optimized wind farm performance. In addition to these two comparisons, another comparison should be done on the optimized control strategies comparing differences between the PARK model and the Mosaic Tile model.

Figures 4.1, 4.2, and 4.3 show the results comparing the optimized wind farm control strategies between the Mosaic Tile model and the PARK model for a wind farm with three, five, and ten turbines in a row, respectively. The standard

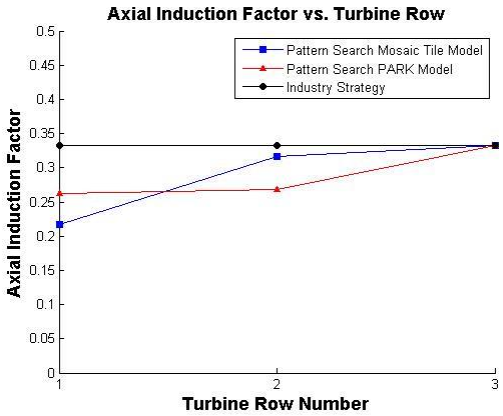



**Figure 4.1:** Wake model comparison of axial induction factor configurations for a 3 turbine row.

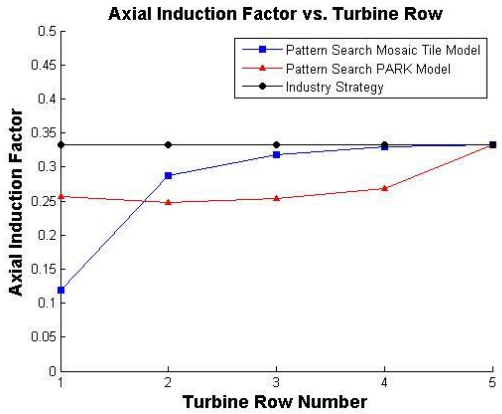



**Figure 4.2:** Wake model comparison of axial induction factor configurations for a 5 turbine row.

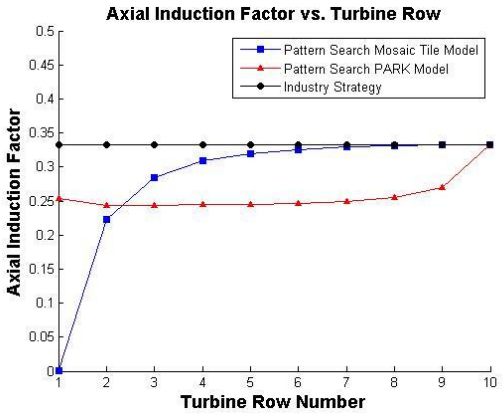



**Figure 4.3:** Wake model comparison of axial induction factor configurations for a 10 turbine row.

industry strategy where each turbine operates at its own optimum (i.e. each turbine operating at an axial induction factor of 1/3) is also shown in these figures for a baseline comparison with what is currently being used in wind farms. The results show that for the PARK model the resulting optimized strategy should have the last turbine in the row operate at its maximum ($a$ = 1/3). This is to be expected as there are no wind turbines behind it that will be affected by its wake, so it should extract as much energy out of the wind as it can. For the turbines in the middle we see that from the back row the axial induction factor should drop to just below 0.25, and the front turbine should be just above 0.25. This trend remains fairly consistent for the PARK model regardless of the number of rows of wind turbines in the wind farm. Results of the velocity deficits for the ten turbine wind farm are also given in Figure 4.4 and show that there is a drop from the first turbine to the second turbine, however it is not as much as the industry strategy. Beyond the second row the velocity deficit remains fairly constant, where as the industry strategy continues to drop slightly.

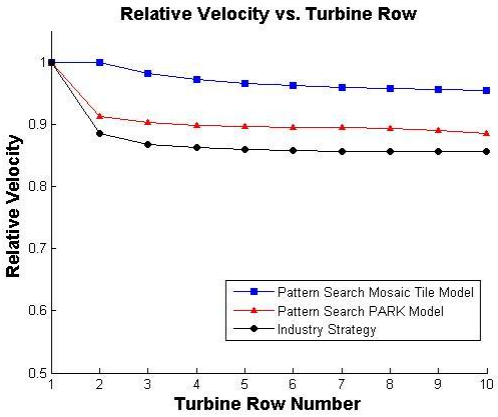



**Figure 4.4:** Wake model comparison of the velocity deficits for a 10 turbine row.

The Mosaic Tile model shows much different results when compared to the PARK model as seen in Figures 4.1, 4.2, and 4.3. Similar to the PARK model, the optimized control strategy with the Mosaic Tile model shows that the last turbine should operate at an axial induction factor of 1/3. However the results for the remaining upwind turbines are much different. In each case the axial induction factor continues to drop for each turbine as the number of downwind turbines increases. With the ten turbine row we see that the front turbine should operate at an axial

induction factor of zero. This essentially means that this turbine should not be running at all. If the remaining nine rows are then optimized with the front row removed it would then result in the second row being shut off. This leads to a contradiction as continuing to repeat this process will result with a final solution dictating that all of the turbines should operate at an axial induction factor of zero.

One possible reason to explain these results is how the Mosaic Tile model handles wake interactions. Unlike the PARK model that uses a superposition method to account for wake interactions (see equation 2.14), the Mosaic Tile model accounts for wake interactions using an added wake expansion term (see equation 2.16). This means that the velocity of a wind turbine depends on the wake effect from the turbine closest upstream to it instead of a combination of all the turbines in front of it. This results in the optimization method determining that each turbine should operate less efficiently than the turbine behind it. Frandsen notes that further development of the Mosaic Tile method includes investigating how the wake depends on turbine operational characteristics (including the axial induction factor) as well as wake overlapping effects [9]. Hence the results presented here from the Mosaic Tile model require further investigation.

**Table 4.1**: Wake model comparisons of percent power increase for three different wind farm sizes.

| Wind Farm Size | Mosaic Tile | PARK |
|---|---|---|
| 3 | 4.52% | 2.61% |
| 5 | 11.2% | 4.09% |
| 10 | 27.6% | 5.62% |

Comparing the overall performance increases of the two wake models shows that the Mosaic Tile model produces a greater percent increase in power

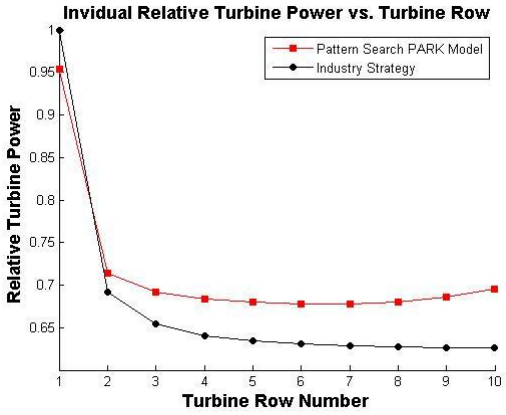



**Figure 4.5:** Power produced at each turbine for the optimized method compared to the traditional method.

when compared to the PARK model as seen in Table 4.1. However, as mentioned before the results from the Mosaic Tile model require further investigation. Figure 4.5 shows more detail of the PARK model results with the power produced at each turbine. Here the turbine's power is being normalized by the power produced by a wind turbine operating at peak performance absent of any wake effects. For the optimized wind farm we see that while the first turbine doesn't produce as much power, the rest of the turbines produce more power when compared to the traditional wind farm operational method. Figure 4.6 shows the sum of the power produced at each row normalized by the power produced of the entire wind farm using a traditional operational method. For instance, turbine row 5 is the sum of the power produced from the first 5 rows of turbines divided by the total power produced of the traditional wind farm. This plot shows that for the first two rows of turbines the power produced with the traditional method is greater than the optimized method. However with more turbines added behind the first two rows the optimized method yields more power produced than the traditional method.

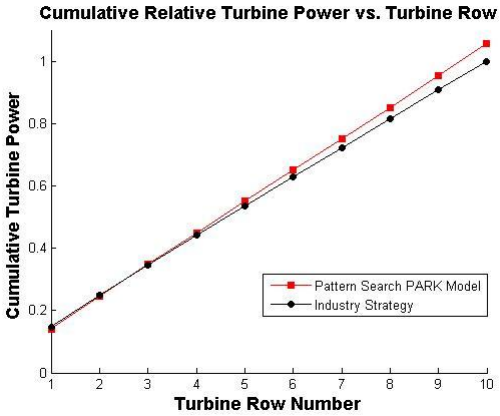



**Figure 4.6:** Cumulative power produced for a given turbine row comparing the optimized method to the traditional industry strategy.

Comparisons were also made with the work done at the Energy research Centre of the Netherlands [5, 38]. First a comparison between the analytical model proposed by the ECN with the

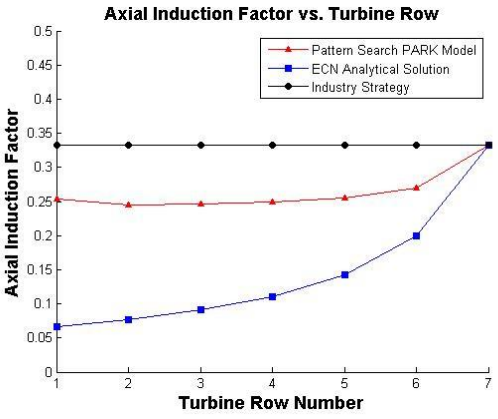



**Figure 4.7:** Comparison of the axial induction factors between the industry strategy, the optimized strategy, and the ECN analytical strategy.

optimized control strategy was made and can be seen in Figure 4.7. Here the analytical model shows that the turbines should operate at a lower axial induction factor than what the optimized PARK model results produced. A similar pattern of axial induction factors exists between these two results. However using the ECN analytical solution the power produced is less than the power produced than the traditional operational strategy as seen in Table 4.2.

**Table 4.2**: Comparisons of the wind farm performance between the traditional strategy, the optimized strategy, and the ECN strategy.

| Wind Farm Operation | Power Produced | Percent Increase |
|---|---|---|
| Industry Strategy | 4.56 MW | 0% |
| Optimized Strategy | 4.78 MW | 4.92% |
| ECN Strategy | 3.68 MW | -19.36% |

In addition to comparing the optimized results with the analytical solution, an effort has been made to create a simulation environment to reproduce the experimental results presented in Figure 2.13 for an 8 m/s wind speed. From [38] the turbine diameter was set to 25 cm, with a hub height of 0.265 m, and a surface roughness of $0.24 \cdot 10^{-6}$ m. Here the front two turbines were optimized while the rest remained fixed at an axial induction factor of 1/3. The turbine spacing was not specified, so several turbine spacings were investigated. Figure 4.8 shows that by optimizing the first two turbine rows the axial induction factors should be between 0.22 and 0.27 depending on the turbine spacing. This configuration most likely corresponds to a "Heat and Flux" unit of 2. Figure 4.9 shows the percent power increase for the different turbine

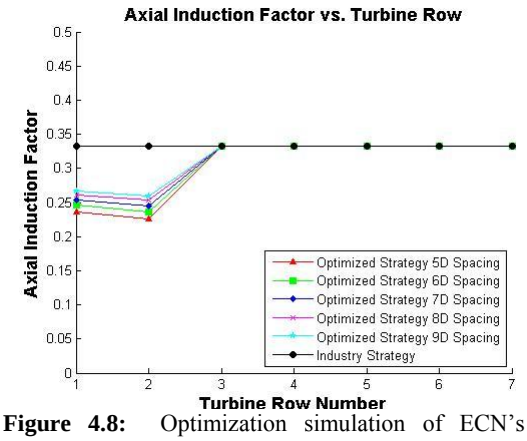



**Figure 4.8:** Optimization simulation of ECN's empirical wind tunnel experiments.

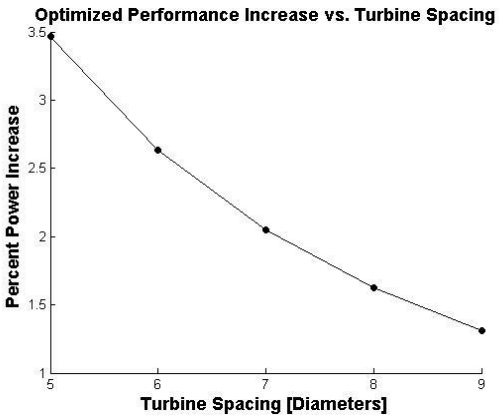



**Figure 4.9:** Percent power increase of an optimization simulation of ECN's empirical wind tunnel experiments

spacings ranges from ~1.2% to ~%3.5 depending on the turbine spacing. With a closer wind farm spacing a larger power increase can be obtained. This is to be expected since the wake loss effect is more significant for turbines spaced closer together. ECN's results showed a power increase of about 2% for an 8 m/s wind speed with a "Heat and Flux" unit of 2 which is in good agreement with the 2.05% increase for a 7 diameter turbine spacing presented here.

### 4.1.2 Wind Farm Size and Number of Rows Controlled

An investigation has also been done to show how varying the number of turbines in the wind farm affects how much the overall performance increase could be improved. Additionally a comparison was made to show how much the performance could be improved by increasing the number of turbines optimized in a wind farm.

In comparing how much of an effect the size of the wind farm has on the attainable improvement in performance a simulation has been done where each turbine's axial induction factor was allowed to vary. The optimized results were then compared to the baseline case where all turbines were operating at an axial induction factor of 1/3 in order to compute a percent power increase to evaluate the amount of performance improvement. Figure 4.10 shows that from varying the size of the wind farm from 1 turbine in a row to 20 turbines in a row, the optimization method continues to increase the overall performance of the wind farm. Initially the performance increase jumps rapidly up to ~4% from a 1 to 5 sized wind farm row. Then from 5 to 20 turbines the performance increase only increase by another 2% making the overall performance increase ~6%. This is because with more wind turbines the wake loss becomes less of an issue as most of

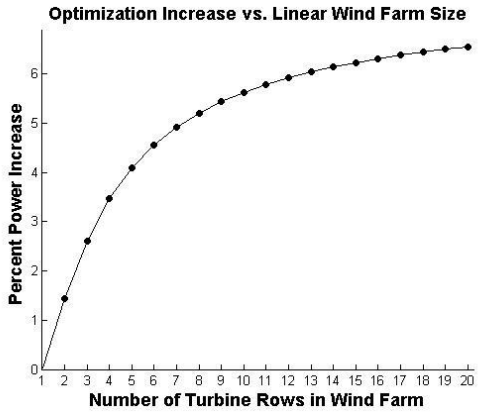



**Figure 4.10:** Percent power increase for varying wind farm sizes.

the velocity deficit occurs between the first and second turbine, while the rest remain fairly constant due to increased turbulent mixing from multiple wakes.

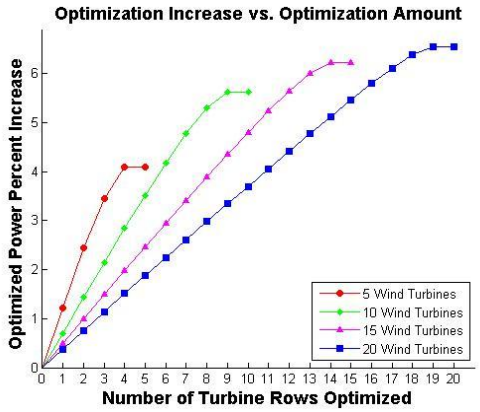



**Figure 4.11:** Percent power increase for varying wind farm sizes as well as number of turbine rows optimized.

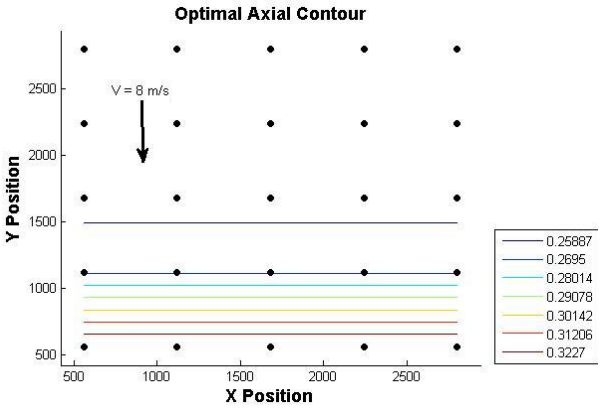
A comparison was also done to see how varying the number of wind turbines controlled with the optimized strategy would affect the overall performance. Here the wind farm size would remain fixed. Then the performance of the wind farm was evaluated for the first row being optimized and the other rows remaining at a fixed axial induction factor of 1/3. Then the performance was assessed for the case where the first two rows were optimized and the subsequent rows remained fixed. This process was repeated until all the wind turbines axial induction factors were optimized. Varying the number of wind turbines that were optimized showed that by increasing the number of wind turbines allowed to be optimized the overall wind farm performance would also increase. Figure 4.11 shows that this is mostly a linear result. Since the optimization method always converged to the last turbine operating at an axial induction factor of 1/3 (which is what it was already being fixed to) there is no difference in performance increase between the case where all the rows were optimized and the case where all but the last row were optimized. Additionally it can be seen in Figure 4.11 that the rate of change of the performance increase vs. the number of rows controlled has a steeper slope for smaller wind farm sizes than for larger ones. This is because the jump from 1 to 2 turbines that are being optimized is a much larger fraction when compared to the wind farm size for a wind farm with 5 turbines than a wind turbine with 20 turbines.

## 4.2 Wind Farm Array Results

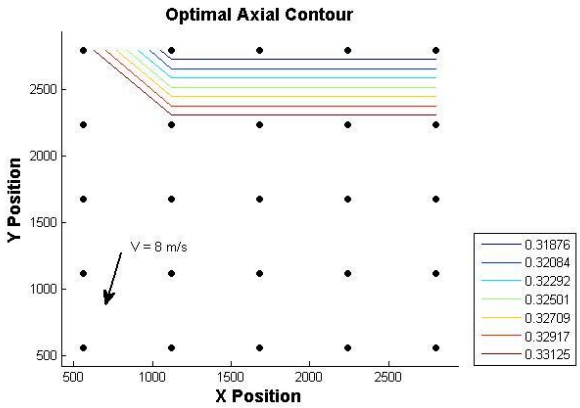
In addition to the results of a linear wind farm, investigation of two dimensional array wind farms has also been performed. The results from these wind farms are presented here. First the results for a simple square lattice as well as a hexagonal lattice are given. Further investigation has been done to determine an optimization strategy for the Horns Rev wind farm. Additionally the optimization algorithm has been applied to a layout proposed by Mosetti where the wake effect has already been minimized through a layout optimization method.

### 4.2.1 Square Lattice with Rotating Wind Direction

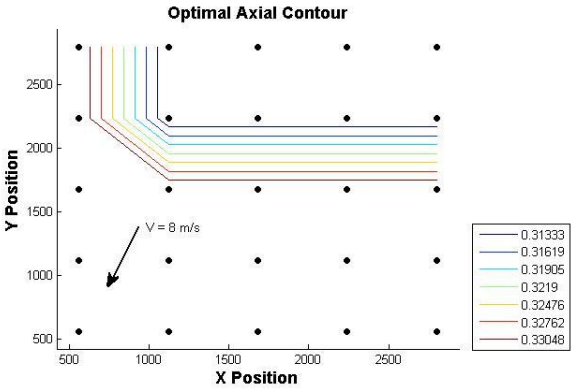
In investigating the optimization results for two dimensional arrays the simplest layout to investigate is a square lattice. For the square lattice a five by five wind farm has been created with a seven diameter spacing between each turbine in both the x and the y directions. Offshore conditions where used with a surface roughness of $10^{-4}$. Part (a) of Figure 4.12 shows a contour plot of the axial induction factors for each wind turbine for a north (0°) wind. Like the linear wind farm the last row of turbines optimizes to an axial induction factor of 1/3 and the rest are around 0.25. Figure 2.12 parts (b) through (i) show how the axial induction factor configuration changes as the wind rotates from a north wind to an east wind. Since the offshore surface roughness was $10^{-4}$ the wake spreading angle was 2.13°. This meant that the wakes line up with other turbines only for specific angles. In cases where the wind direction did not line up with the wind farm layout the resulting in the optimized control strategy conveged to the industry strategy (i.e. each turbine operating at an axial induction factor of 1/3) with no performance improvement. This made sense since the absence of wake losses due to the wind direction misalignment would result in each turbine extracting as much energy as it could out of the wind.



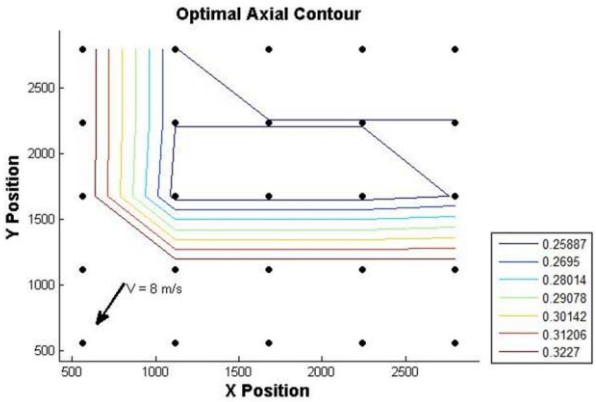

**(a) 0° Wind Direction**



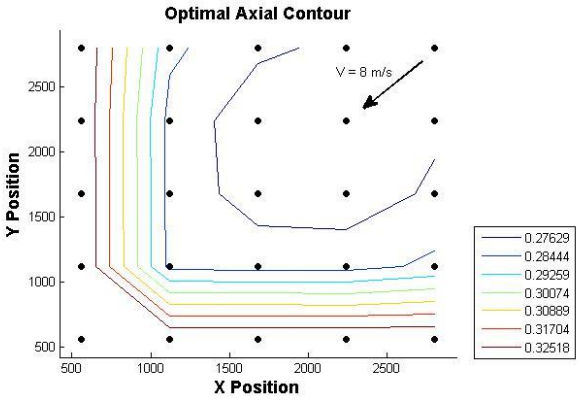

**(b) 14.04° Wind Direction**



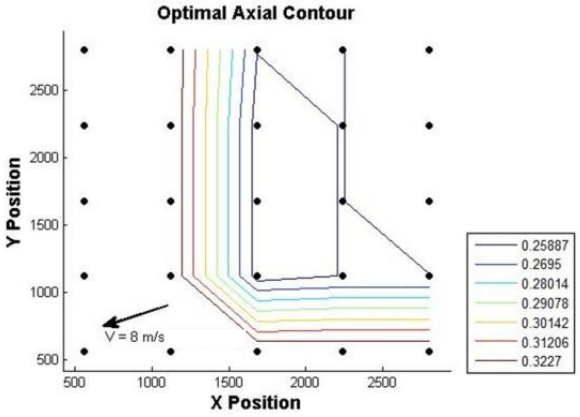

**(c) 18.43° Wind Direction**



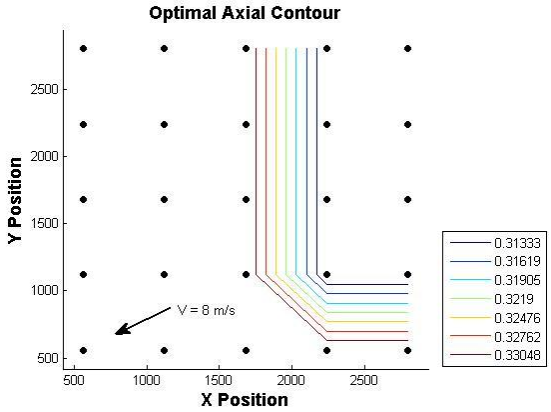

**(d) 26.57° Wind Direction**



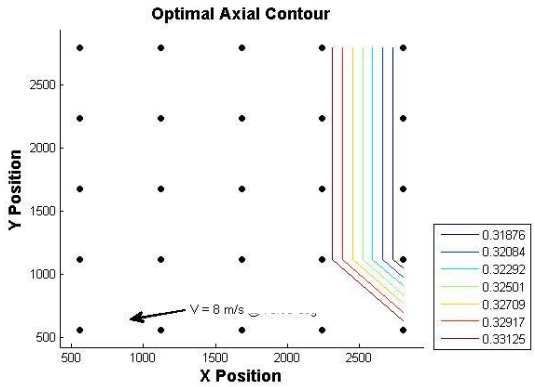

**(e) 45° Wind Direction**



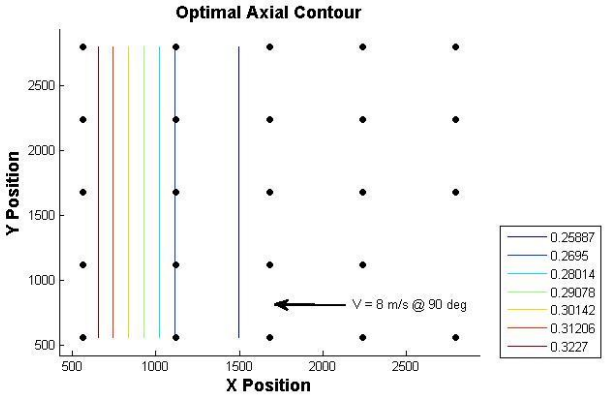

**(f) 63.43° Wind Direction**




**(g) 71.56° Wind Direction**




**(h) 75.96° Wind Direction**




**(i) 90° Wind Direction**

**Figure 4.12 (a) through (i):** Contour map of the axial induction factor configurations for a five by five square lattice with different wind directions.

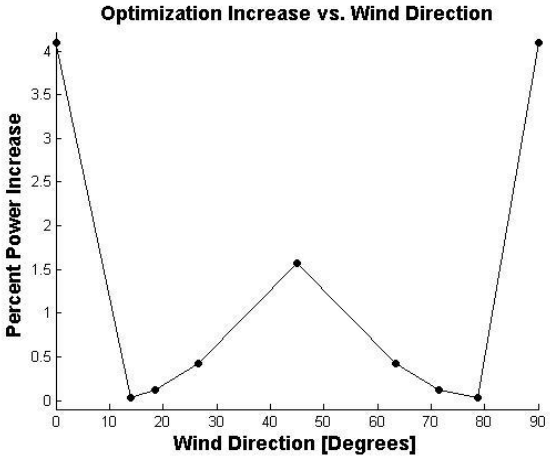



**Figure 4.13:** Percent power increase of the square lattice wind farm vs. wind direction. The optimization method has the largest impact when the wind is more aligned with the wind farm layout.

Putting together the performance results of the square lattice wind farm for different wind directions resulted in the performance curve in Figure 4.13. Here we see that when the wind direction is aligned with the wind farm layout the largest improvement can be achieved. When the wind farm is slightly aligned (for a wind direction of 11.31° and 78.69°) only a slight improvement can be obtained. This is to be expected though as most of the wind turbines are unaffected by the wakes of the other turbines and optimize to $a = 1/3$ which is the same as the industry strategy.

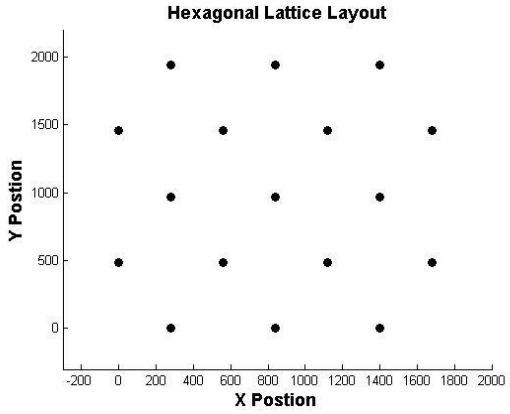



**Figure 4.14:** Turbine layout for the hexagonal lattice wind farm.

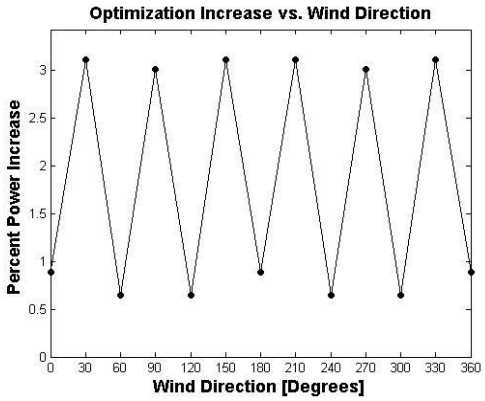



**Figure 4.15:** Percent power increase for a hexagonal lattice wind farm vs. wind direction. Like the square lattice the optimized strategy has the largest impact when the wind direction is aligned with the turbine layout.

### 4.2.2 Hexagonal Lattice Wind Farm

The results of the square lattice showed that the performance of the wind farm can be most improved when the wind direction was aligned with the turbine layout. A hexagonal lattice (see Figure 4.14) was also investigated and showed similar results to the square lattice as seen in Figure 4.15. Here we see that the performance increase varies from ~1% to ~3% for different 30° interval wind directions. The larger performance increase is achieved when the wind direction is aligned with rows with the smallest turbine spacing, where as the other angles the turbine spacing is larger and the wake losses are less of a factor.

### 4.2.3 Horns Rev Wind Farm Optimization

In considering the optimization of an existing wind farm the Horns Rev wind farm was chosen as the geometrical spacing was readily available. Three different optimizations were performed each with a different wind direction that was aligned with rows of wind turbines with three different turbine spacings: 7 diameters, 9.4 diameters, and 10.5 diameters. The performance improvements from each optimization can be seen in Table 4.3. Additionally the optimized axial induction factor configurations have been shown in the contour maps of Figures 4.17, 4.18, and 4.19.

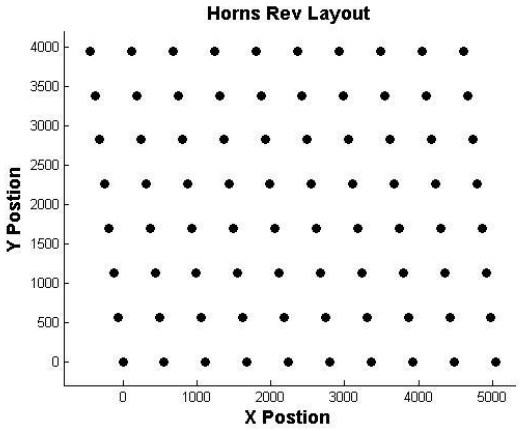



**Figure 4.16:** Turbine layout of the Horns Rev wind farm [11].

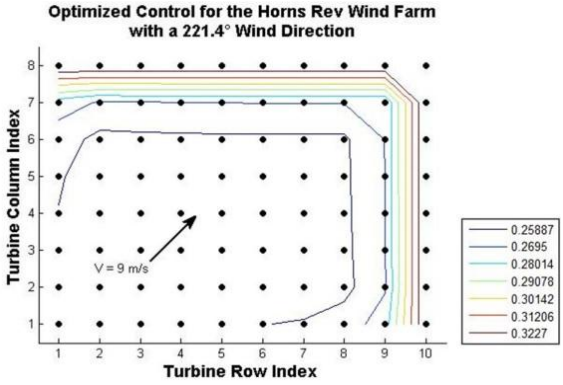



**Figure 4.17**: Optimized axial induction factor configuration of the Horns Rev wind farm for a 221.4° wind direction.

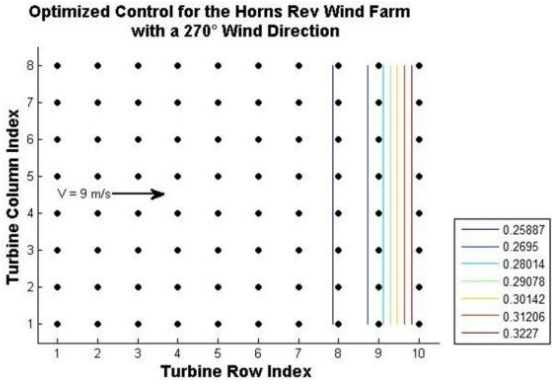



**Figure 4.18**: Optimized axial induction factor configuration of the Horns Rev wind farm for a 270° wind direction.

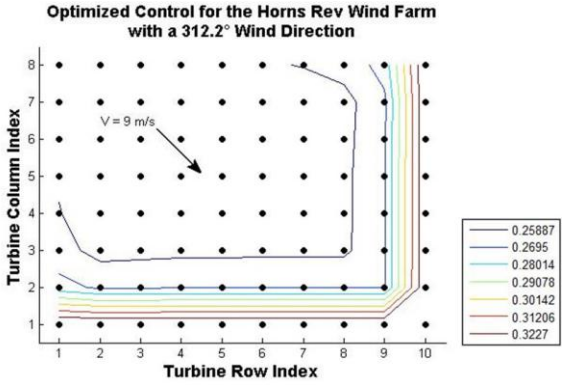



**Figure 4.19**: Optimized axial induction factor configuration of the Horns Rev wind farm for a 312.2° wind direction.

**Table 4.3**: Comparisons of the Horns Rev wind farm performance to three different wind directions and turbine spacings.

| Wind Direction | Turbine Spacing | Performance Increase |
|---|---|---|
| 221.4° | 9.4 Diameters | 2.53% |
| 270° | 7 Diameters | 5.62 % |
| 312.2° | 10.5 Diameters | 2.06 % |

In these figures the turbines are shown at their row and column index locations. However the results were obtained using the actual geometry of the wind farm as seen in Figure 4.16.

The results from the optimization of the Horns Rev wind farm show that the turbines in the back of the wind farm should be optimized to an axial induction factor of 1/3. Then for the remainder of the turbines the axial induction factor should drop to a value of 0.24 to 0.27 depending on how many turbines are located behind it. This strategy remains the same, but rotated with the different wind directions. It is also interesting to note that the performance increase is the greatest (5.62%) for the direction where the turbine spacing is the smallest (7 diameters). Conversely the improvement is the smallest (2.06%) for the direction where the turbine spacing is the greatest (10.5 diameters). Since the Horns Rev wind farm follows a parallelogram shaped lattice that is almost a square lattice shape these results are to be expected as similar results were found earlier with the square lattice wind farm optimization.

### 4.2.4 Optimized Layout and Wind Farm Control

Taking the results from Mosetti's layout optimization study [34] for a constant wind speed and direction the operational optimization algorithm was then applied to determine if further improvements could be made on the overall performance of the wind farm. Refer to Figure 3.4 for details of the turbine layout. By optimizing the axial induction factors the net power produced increased from 20.04 MW to 20.05 MW resulting in a 0.034% increase. Hence if a wind farm has already been optimized with its layout configuration optimizing the axial induction factors has minimal performance improvements. This is because the wind farm has already been optimized to minimize the wake loss effects.

# Chapter 5

# Conclusions

It was shown through this project that by optimizing the axial induction factors of a wind farm the losses due to wind turbine wakes can be minimized. While it was shown that this results in less power being produced by the turbines on the windward side of the farm, more wind was allowed to pass through so that the turbines on the leeward side of the farm can produce more power increasing the overall power produced by the entire wind farm. Using this idea the performance increase was evaluated to be about 4% to 6% in the total power produced depending on the wind farm size, turbine spacing, and number of rows controlled.

It was also shown in this research project that the axial induction factor configuration did not depend on which optimization method was used. It did however depend greatly on which wake model was used especially when dealing with how the different models accounted for wake interactions. With this in mind, the PARK model was determined to be sufficient in determining the optimized control strategies for wind farms. The resulting axial induction factors for a linear wind farm showed that by lowering the axial induction factors of the windward side of the wind farm from 1/3 to about 0.25 for most of the turbine layouts explored, an overall power increase could be obtained.

Comparisons were also made with the Energy research Centre of the Netherlands empirical wind tunnel test and showed that similar results could be achieved through the methods used in this research project. The wind tunnel experiments showed a performance increase of about 2% where this project's results showed a 2.05% increase.

Results from a two dimensional square lattice wind farm showed how the operational strategy should change as the wind direction rotates around the wind farm. It also showed that the amount of performance increase that could be achieved depended on how much the wind direction was aligned with the wind turbine rows. If the wind direction was more aligned, a greater performance increase could be achieved since the wake loss effects were much greater. A hexagonal lattice was also explored and showed similar results to the square lattice with higher performance increases obtained when the wind direction was aligned with lattice directions containing smaller wind farm spacings.

Optimization results for the Horns Rev wind farm showed that the performance increase was the greatest for wind directions where the turbine spacing was the smallest. Additionally, it was found that the axial induction factor configuration followed similar patterns to the square lattice results which were to be expected as the layout of the Horns Rev wind farm is essentially a square lattice.

An optimization was also done for a wind farm layout that was determined from a previous optimization to minimize the cost of energy. The performance improvements using an optimum control strategy was evaluated to be only a 0.034% increase. This was to be expected as the layout of the wind farm was already optimized to minimize the wake loss effects.

There are many ways to further explore the results that have been investigated throughout this research project. These ideas could be used to further extend this research in order to be able to run faster with greater precision, and have a more practical application towards wind farms. Some of the concepts in mind for future work for this research include the following ideas to extend the scope of this project:

- Implement the methods in a C++/FORTRAN environment in order to improve the processing performance and allow for the code to be run on a parallel machine
- Extend the optimization to include different and more complex models (i.e. the eddy-viscosity model) to compare results and to be able to take into account further physical complexities (turbulence, atmospheric stability, complex terrain, etc.)
- Combine a layout optimization with a control optimization to determine if it is economically favorable to place more turbines closer together in a wind farm with an optimized control strategy
- Apply the optimal control strategy presented here to an existing wind farm (i.e. Horns Rev) in order to get empirical data to compare the results to
- Explore new methods to evaluate different types of wake interactions
- Extend the software to include WT_Perf [42] to model the wind turbine so that the pitch may be optimized instead of the axial induction factor
- Investigate how the control strategies presented here change with higher wind speeds where some of the wind turbines will operate in region III
- Determine control methods to actuate how a wind farm controls its performance to match the results being presented in this project

As mentioned before the axial induction factor configurations were dependent on the wake model that was used. Furthermore the power performance results presented through this project were also dependent on which wake model was used. The 4% to 6% increase in power performance reported here may under or over predict the performance results when compared to other wake model results, but the trends that have been shown here should remain the same (see Figure 2.10). These trends include the following:

- The performance increase reduces with larger turbine spacing (see Figure 4.9)
- How the performance increase varies with the wind farm size (see Figure 4.10)
- How the performance increase varies as the number of turbine rows optimized in a wind farm (see Figure 4.11)
- The axial induction factor configuration pattern should rotate accordingly with a changing wind direction (see Figure 4.12)
- The performance increase is larger when the wind is more aligned with the turbine rows of the wind farm (see Figure 4.13)

Implementing the optimization techniques used in this research with other wake models will provide comparisons that can be used to determine how the performance results vary by taking other effects into account. However it is also necessary to compare these results to empirical data from existing wind farms. With this it could then be determined if the different models under or over predict actual performance results. Collecting data from existing wind farms using an optimized control strategy is an area of future research that would be very useful to further validate the results of this research project.

By using the results being presented through this research wind farms should be able to produce 4% to 6% more power depending on the wind farm size, turbine spacing, and the number of wind turbines controlled. This will result in a lower cost of energy which is desired not only by energy consumers, but by wind farm developers, and energy providers as well. Another benefit from using an optimized axial induction factor configuration is that the fatigue loads from the wind experienced in the wind turbine blades and tower will be reduced since lowering the axial induction factors will reduce the amount of turbulence created in the downstream wake. In addition to the lower cost of energy and reduction in loading these

results will also increase the amount of green energy that wind farms can produce and further offset energy that is being created by unsustainable methods such as fossil fuels and nuclear power. This will bring us closer to reaching sustainable goals such as the 20% of U.S. electricity from wind power by 2030 and decrease the impact that our energy usage is having on earth.

## References:

[1] D. Lashof, D. Ahuja. Relative contributions of greenhouse gas emissions to global warming. *Nature* 1990 **344**: 529-531.

[2] US Department of Energy, "Wind Energy by 2030: Increasing Wind Energy's Contribution to U.S. Electricity Supply", 2008, http://www.nrel.gov/docs/fy08osti/41869.pdf, accessed March 23, 2011.

[3] U.S. Energy Information Administration, "Annual Energy Outlook 2011 Early Release Overview", http://www.eia.gov/forecasts/aeo/pdf/0383er(2011).pdf, accessed March 23, 2011.

[4] S. Pacala, R. Socolow. Stabilization Wedges: Solving the Climate Problem for the Next 50 Years with Current Technologies. *Science* 2004 **305**: 968-972.

[6] R. Froude. On the part played in propulsion by difference of fluid pressure. *Trans. Institution of Naval Architects* 1889 **30**: 390-405.

[7] J. Manwell, J. Mcgowan, A. Rogers. *Wind Energy Explained: Theory, Design, and Application*. 2nd ed. United Kingdom: John Wiley & Sons Ltd., 2009.

[8] L. Vermeer, J. Sørensen, A. Crespo. Wind turbine wake aerodynamics. *Progress in Aerospace Sciences* 2003 **39**: 467-510.

[9] S. Frandsen, *et al.* The Making of a Second-generation Wind Farm Efficiency Model Complex. *Wind Energy* 2009 **12**: 445-458.

[10] R. Barthelmie, *et al.* Modelling and Measurements of Power Losses and Turbulence Intensity in Wind Turbine Wakes at Middelgrunden Offshore Wind Farm. *Wind Energy* 2007 **10**: 517-528.

[11] R. Barthelmie, L. Jensen. Evaluation of wind farm efficiency and wind turbine wakes at the Nysted offshore wind farm. *Wind Energy* 2010 **13**: 573-586.

[12] R. Barthelmie, *et al*. Modelling the impact of wakes on power output at Nysted and Horns Rev. presented at *European Wind Energy Conference* Marseilles, France, 2009.

[13] R. Barthelmie. Wind Turbine Wakes Virtual Laboratory. [Online], Wind Turbine Wake Virtual Laboratory Wakes Site through www.oncourse.iu.edu, accessed March 28, 2011.

[14] A. Crespo, J. Hernández, S. Frandsen. Survey of Modelling Methods for Wind Turbine Wakes and Wind Farms. *Wind Energy* 1999 **2**: 1-24.

[15] P. Lissaman. Energy Effectiveness of Arbitrary Arrays of Wind Turbines. Presented at *American Institute of Aeronautics and Astronautics Conference* Pasadena, CA, 1979.

[16] N. Jensen. A Note on Wind Generator Interaction. Roskilde, Denmark: Riso National Laboratory; 1983. Technical report Riso-M-2411.

[17] I. Katić, J. Højstrup, N. Jensen. A Simple Model for Cluster Efficiency. presented at *European Wind Energy Conference* Rome, Italy, 1986.

[18] A. Kusiak, Z. Song. Design of wind farm layout for maximum wind energy capture. *Renewable Energy* 2010 **35**: 685-694.

[19] S. Frandsen. On the wind speed reduction in the center of large clusters of wind turbines. *Journal of Wind Engineering and Industrial Aerodynamics* 1992 **39**: 251-265.

[20] O. Rathmann, R. Barthelmie, S. Frandsen. Turbine Wake Model for Wind Resource Software. presented at *European Wind Energy Conference* Athens, Greece, 2006.

[21] S. Frandsen, *et al.* Analytical Modelling of Wind Speed Deficit in Large Offshore Wind Farms. *Wind Energy* 2006 **9**: 39-53.

[22] J. Ainslie. Calculating the Flowfield in the Wake of Wind Turbines. *Journal of Wind Engineering and Industrial Aerodynamics* 1988 **27**: 213-224.

[23] T. Stovall, G. Pawlas, P. Moriarty. Wind Farm Wake Simulations in OpenFOAM. presented at *48th AIAA Aerospace Sciences Meeting*, Orlando, Florida, 2010.

[24] R. Mikkelsen, J. Sørensen, S. Øye, N. Troldborg. Analysis of Power Enhancement for a Row of Wind Turbines Using the Actuator Line Technique. *Journal of Physics: Conference Series, The Science of Making Torque from Wind* 2007 **75**: 1-8.

[25] R. Barthelmie, et. al. Flow and wakes in large wind farms in complex terrain and offshore. presented at *European Wind Energy Conference* Brussels, Belgium, 2008.

[26] A. Crespo, J. Hernández, E. Fraga, C. Andreu. Experimental Validation of the UPM Computer Code to Calculate Wind Turbine Wakes and Comparison with other Models. *Journal of Wind Engineering and Industrial Aerodynamics* 1988 **27**: 77-88.

[27] A. Duckworth, R. Barthelmie. Investigation and Validation of Wind Turbine Wake Models. *Wind Engineering* 2008 **32**: 459-475.

[28] C. Beightler, D. Phillips, D. Wilde. *Foundations of Optimization*, 2nd ed. Englewood Cliffs, NJ, Prentice-Hall, Inc., 1979..

[29] J. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*, 2nd ed. Boston, MA, MIT Press, 1992.

[30] M. Affenzeller, S. Winkler, S. Wagner, A. Beham. *Genetic Algorithms and Genetic Programming: Modern Concepts and Practical Applications*, 1st ed. Boca Raton, FL, Chapman & Hall/CRC Taylor and Francis Group, 2009.

[31] N. Metropolis, S. Ulam. The Monte Carlo Method. *Journal of the American Statistical Association* 1949 **44**: 335-341.

[32] R. Hooke, T. Jeeves. "Direct Search" Solution of Numerical and Statistical Problems. *Journal of the Association for Computing Machinery* 1961 **8**: 212-229.

[33] V. Torczon. On the Convergence of Pattern Search Algorithms. *SIAM Journal on Optimization* 1997 **7**: 1-25.

[34] G. Mosetti, C. Poloni, B. Diviacco. Optimization of wind turbine positioning in large wind farms by means of a genetic algorithm. *Journal of Wind Engineering and Industrial Aerodynamics* 1994 **51**: 105-116.

[35] S. Grady, M. Hussaini, M. Abdullah. Placement of wind turbines using genetic algorithms. *Renewable Energy* 2005 **30**: 259-270.

[36] G. Marmidis, S. Lazarou, E. Pyrgioti. Optimal placement of wind turbines in a wind park using Monte Carlo simulation. *Renewable Energy* 2008 **33**: 1455-1460.

[37] B. Du Pont, J. Cagan. An Extended Pattern Search Approach to Wind Farm Layout Optimization. presented at *ASME International Design Engineering Technical Conference* Montreal, Canada, 2010.

[38] G. Corten, P. Schaak. More Power and Less Loads in Wind Farms: "Heat and Flux". presented at *European Wind Energy Conference* London, England, 2004.

[5] G. Corten, P. Schaak. Heat and Flux: Increase of Wind Farm Production by Reduction of the Axial Induction. presented at *European Wind Energy Conference* Madrid, Spain, 2003.

[39] L. Machielse, S. Barth, E. Bot, H. Hendriks, G. Schepers. Evaluation of "Heat and Flux" Farm Control. Energy research Centre of the Netherlands. Petten, The Netherlands. Tech. Rep. ECN-E—07-105. 2007.

[40] E. Hau. "The Wind Resource," in *Wind Turbines: Fundamentals, Technologies, Application, Economics*, 2nd ed. Krailling, Germany, Springer, 2006, ch. 13, sec. 3, pp. 459-468.

[41] D. Goldberg. *Genetic Algorithms in Search, Optimization & Machine Learning*. 1st ed. Reading, MA, Addison-Wesley, 1989.

[42] M. Buhl. NWTC Design Codes (WT_Perf). 2011, [Online], wind.nrel.gov/designcodes/simulators/wtperf/ accessed April 5, 2011.

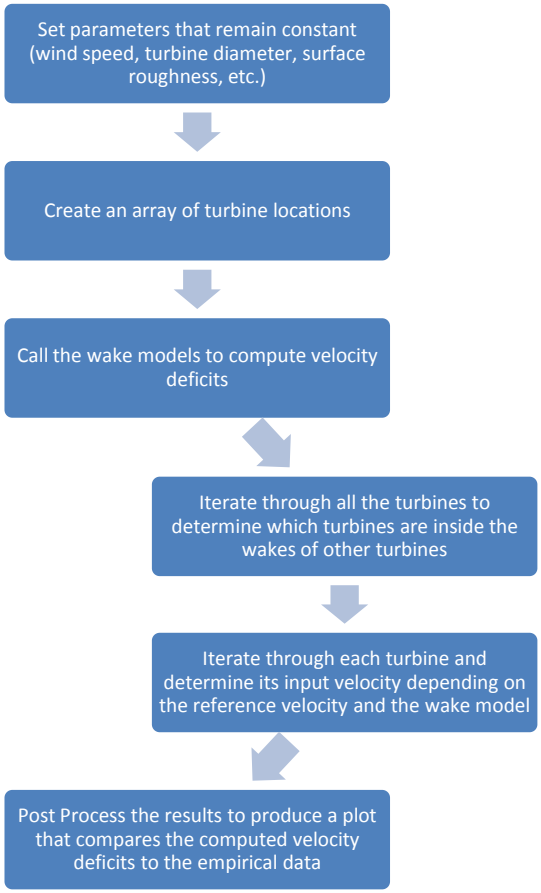
# Appendix A

## Program Flow Diagrams

This appendix contains a high level view of the processes that occur during different simulations that were used throughout this research project. Three separate diagrams are given for each of the main programs that are given in Appendix B.

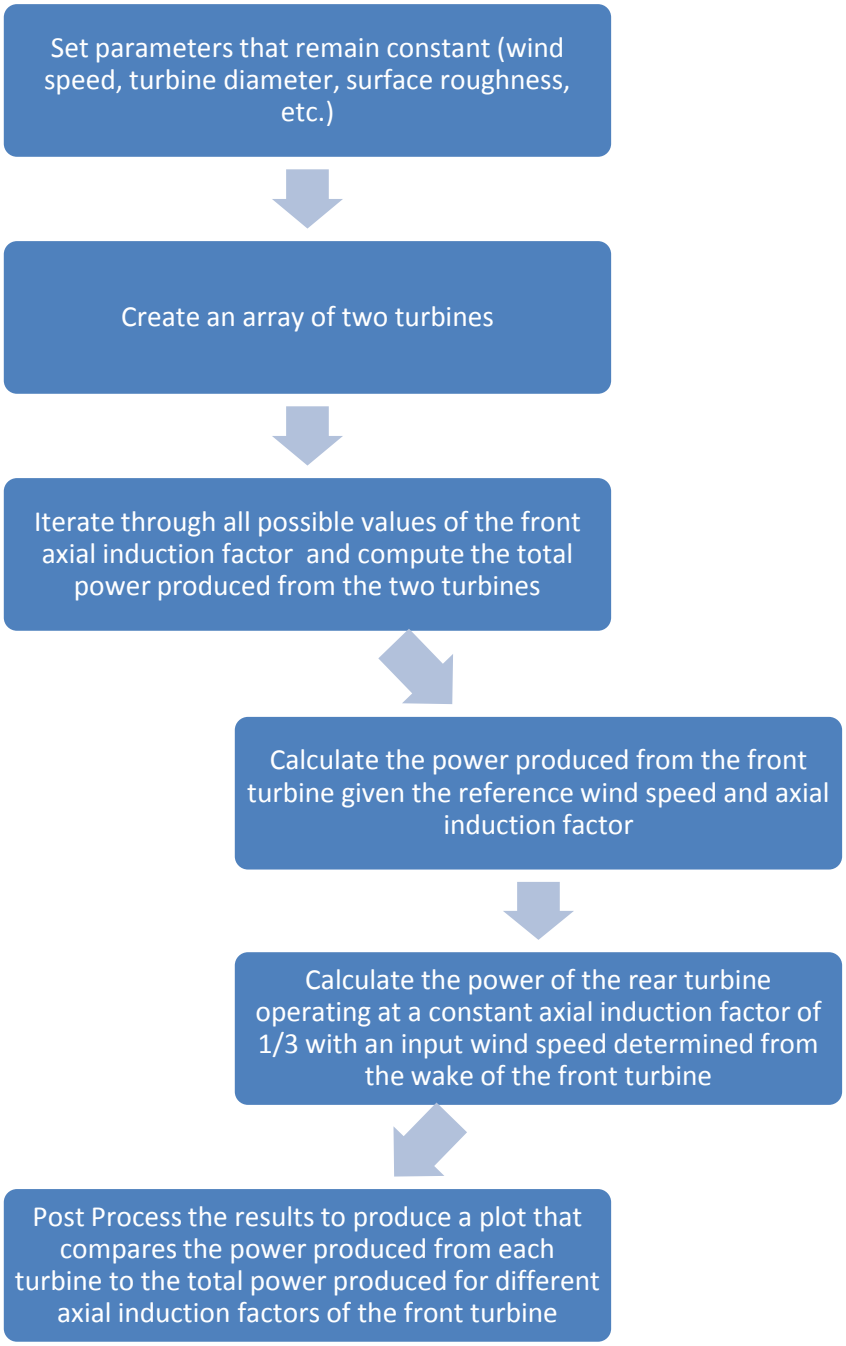
### A.1 Wake Model Validation

Below is a flow diagram for the simulation that was done to check to see that the wake models used throughout this research were in good agreement with empirically observed wake model data. The code corresponding to this diagram can be found in Appendix B.1



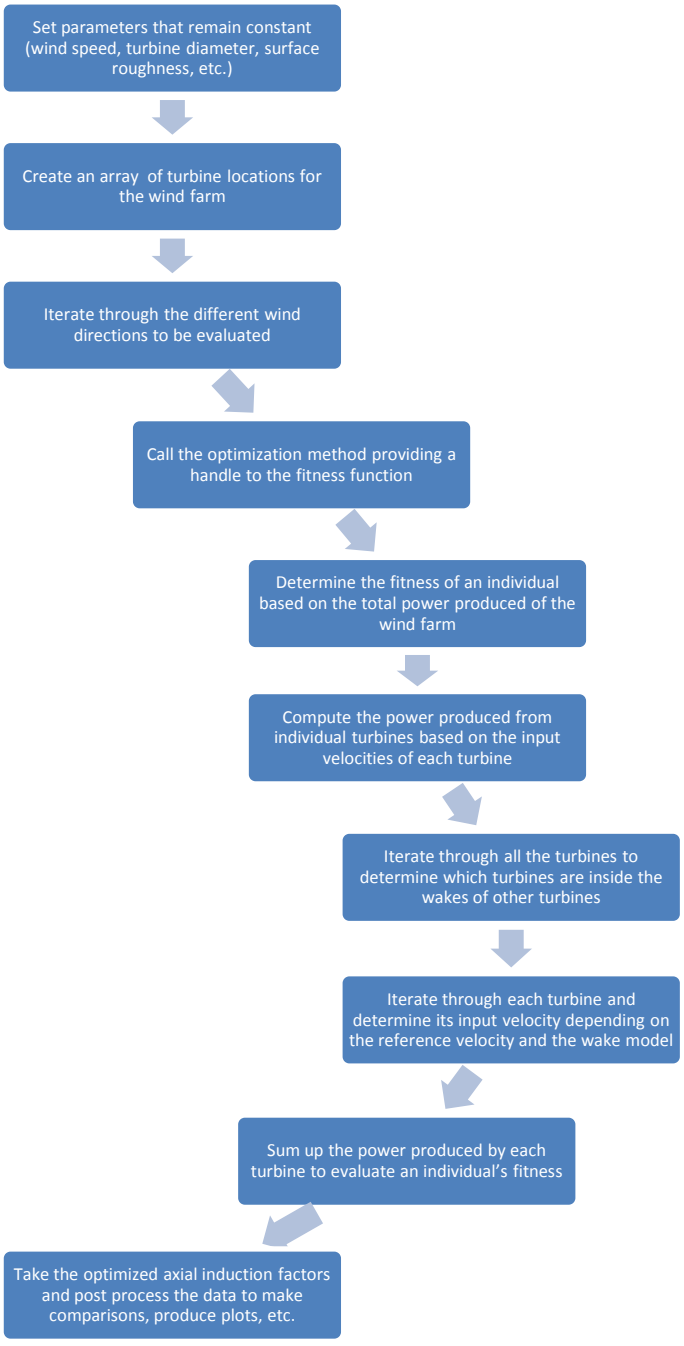
## A.2 Turbine/Wake Coupled Model Validation

Below is a flow diagram for the simulation that was done to check to see that the wake model coupled with the turbine model showed good performance results when compared to a simulation performed by the ECN [5]. The code corresponding to this diagram can be found in Appendix B.2



## A.3 Wind Farm Control Optimization Method

Below is a flow diagram for the simulations that were performed for different wind farm layouts to determine the optimal control strategy to minimize the wake loss effects. The code corresponding to this diagram can be found in Appendix B.3.



# Appendix B

# Code Implementation

This appendix contains the MATLAB code that was implemented throughout this research project. Brief descriptions of each piece of code will be provided in order to describe the basic flow of the code. The code has been organized by starting with the main programs used to start and run the simulations and is followed by the lower level functions that were used for specific turbine, wake and optimization computations.

## B.1 testFindInputVelocities.m

```
function testFindInputVelocities

    %%%   This function calls the two implemented wake models
    %%%   and performs a comparison to the Horns Rev wind farm
    %%%   data in order to validate the models

    clear;
    clc;
    close all;

    %%%   Constants used across different functions
    global kappa windSpeed turbineDiameter windDir

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%   User defined Parameters

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%   Wind Parameters
    windSpeed = 8.0;         % [m/s]
    rho = 1.225;             % [kg/m^3]
    roughness = 1e-4;        % [meters] {from pg. 463 Erich Hau's
```

```
                          % Wind Turbines}
windDir = 270;            % [degrees] defined like a compass
                          % (0 = North, 270 = West) and where
                          % the wind is coming from

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%   Turbine Parameters
turbineDiameter = 80;                   % [meters]
turbineSpacing = 7;                     % diameters
hubHeight = 70;                         % [meters]
gridSize = 7;                           % number of turbines
a_val = 1/3;
axialVals = (a_val)*ones(gridSize,1);  % axial setup

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%   Setup derived variables
kappa = 0.5/( log(hubHeight/roughness) );   % Wake spreading
                                            % coefficient

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%   Initialize turbine locations
%%%   1 x numTurbines east/west grid
turbines = zeros(gridSize,3);
numTurbines = length(turbines(:,1));
for i = 1:numTurbines
    turbines(i,1) = (i-1)*turbineSpacing*turbineDiameter;
end
turbines(:,3) = axialVals;

%%%   Use the wake models to find the velocity deficits:
PARK_VelocityResult = findPARKInputVelocities(turbines);
MosaicTile_VelocityResult = ...
    findMosaicTileInputVelocities(turbines);

%%%   Post Processing:

%%%   Normalize the results by the windSpeed:
MosaicTile_relVel = MosaicTile_VelocityResult./windSpeed;
PARK_relVel = PARK_VelocityResult./windSpeed;

%%%   The next block of code creates a plot
%%%   to compare the wake model computations to
%%%   the Horns Rev Wind Farm Data
figure(1);
hold on;
Horns_Rev_relVel = [1.0065375  0.6756375 ...
```

```
        0.669275    0.63835   0.6261    0.6099375 ...
        0.60065     0.5856125].^(1/3);
    Horns_Rev_Error = [0.012913109 0.039537719    ...
        0.040001562 0.042929677    0.045836729    ...
        0.03762666  0.031876682    0.027283404];
    plot(turbines(:,1),MosaicTile_relVel,'-bo',...
        'MarkerSize',5,'MarkerEdgeColor','b',...
                'MarkerFaceColor','b');
    plot(turbines(:,1),PARK_relVel,'-r^','MarkerSize',...
        5,'MarkerEdgeColor','r',...
                'MarkerFaceColor','r');
    plot(turbines(:,1),Horns_Rev_relVel(1:gridSize),'-ks',...
        'MarkerSize',5,'MarkerEdgeColor','k',...
                'MarkerFaceColor','k');
    errorbar(turbines(:,1),Horns_Rev_relVel(1:gridSize),...
        Horns_Rev_Error(1:gridSize),'k');
    xlabel('Turbine Location','FontSize',14,...
        'FontWeight','bold');
    ylabel('Relative Wind Speed','FontSize',14,...
        'FontWeight','bold');
    title('Input Velocity vs. Turbine Position',...
        'FontSize',14,'FontWeight','bold');
    legend('Mosaic Tile Model','PARK Model',...
        'Horns Rev Data');
    axis([0 3600 0.6 1.05]);
end
```

## B.2 twoTurbineOptimalAxial.m

```
function twoTurbineOptimalAxial

    %%%   This code looks to find the optimal axial induction
    %%%   factor of two wind turbines spaced 7 diameters apart
    %%%   along the wind direction.  It will be used to make
    %%%   A comparison between the performance model used in
    %%%   in this research project with the ECN simulation
    %%%   results.
    %%%
    %%%   This code will use the PARK model to simulate the
    %%%   wakes produced from the wind turbines.

    clear;
    clc;
    close all;

    %%%   Constants used across different functions
```

```
global windSpeed turbineSpacing turbineDiameter ...
    roughness rho hubHeight

%%%   User defined parameters
axialInc = 1e-4;
axialFactors = 0:axialInc:0.5;

%%%   Wind parameters
rho = 1.225;            % [kg/m^3]
windSpeed = 9.0;        % [m/s]


%%%   Turbine parameters
turbineDiameter = 80;   % Diameter of the turbine [meters]
turbineSpacing = 7;     % Diameters
hubHeight = 70;         % Hub Height of the the turbine
                        % [meters]
roughness = 0;          % surface roughness [meters]
                        % chosen to match ECN's study

%%%   Initialize solution vectors
frontPower = zeros(size(axialFactors));
backPower = frontPower;

%%%   compute the solution for the different axial
%%%   induction factors
backAxial = 1/3;
for i = 1:length(axialFactors)
    frontAxial = axialFactors(i);
    frontPower(i) = getPower(frontAxial, windSpeed);
    downWind = getDownWindSpeed(frontAxial);
    backPower(i) = getPower(backAxial, downWind);
end

%%%   Compute the total power produced from
%%%   the two wind turbines
totalPower = frontPower + backPower;

%%%   Normalize the solutions
maxFront = max(frontPower);
frontPower = frontPower./maxFront;
backPower = backPower./maxFront;
totalPower = totalPower./maxFront;

%%%   display the results
figure(1);
```

```
    hold on;
    plot(axialFactors,frontPower,'r-.','LineWidth',1.4);
    plot(axialFactors,backPower,'b--','LineWidth',1.4);
    plot(axialFactors,totalPower,'k','LineWidth',1);
    xlabel('Front Turbine Axial Induction Factor',...
        'FontSize',14,'FontWeight','bold');
    ylabel('Normalized Power','FontSize',14,...
    'FontWeight','bold');
    title('Power Produced from Two Turbines',...
        'FontSize',14,'FontWeight','bold');
    legend('Front Turbine','Back Turbine',...
        'Total Power');

    %%%   determine the maximum power and which axial
    %%%   induction factor produced it
    maxPower = 0;
    maxI = -1;
    for i = 1:length(axialFactors)
        if totalPower(i) > maxPower
            maxI = i;
            maxPower = totalPower(i);
        end
    end
    axialMax = axialFactors(maxI)

    %%%   Determine the power produced from when both
    %%%   turbines were operating at a = 1/3
    for j = 1:length(axialFactors)
        if axialFactors(j) > (1/3) && ...
                axialFactors(j) < ( (1/3) + 2*axialInc)
            powerOneThird = totalPower(j);
        end
    end

    %%%   Compute the percentage of power increased
    percentIncrease = 100*( (maxPower/powerOneThird) - 1 )
end

function uOut = getDownWindSpeed(axial)

    %%%   Determine the downstream wind speed from the
    %%%   wake of the upwind turbine using the PARK Model

    global windSpeed turbineSpacing turbineDiameter ...
        roughness hubHeight
```

```
    alpha = 0.5/( log(hubHeight/roughness) );
    xDist = turbineSpacing*turbineDiameter;
    r_1 = (turbineDiameter/2).*sqrt( (1 - axial)/...
        (1 - 2.*axial) );
    uOut = windSpeed.*( 1 - ( 2.*axial./((1 + ...
        alpha.*xDist./r_1 ).^2) ) );
end

function pOut = getPower(axial, windInput)

    %%%   Get the power produced as a function of pitch,
    %%%   and windspeed

    global rho turbineDiameter

    C_p = 4.*axial.*((1 - axial).^2);
    area = (turbineDiameter.^2).*pi./4;
    pOut = C_p.*rho.*(windInput.^3).*area./2;
end
```

## B.3 optimalWindFarmComputation.m

```
function optimalWindFarmComputation

    %%%   This function is an example of the main source code
    %%%   used for all of the specific simulations done
    %%%   throughout the research.  It can be tailored to
    %%%   run for different wind farm set ups accordingly.
    %%%
    %%%   Below an example is given for the optimization
    %%%   of the Horns Rev Wind Farm

    clear;
    clc;
    close all;

    %%%   Constants used across different functions
    global kappa turbineDiameter windSpeed windDir ...
        turbineLayout rho modelNumber numFixed

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %%%   User defined Parameters

    %%%   Set the model number to 1 for the PARK model
    modelNumber = 1;
```

```
%%%   numFixed set to 0 to optimize the entire wind farm.
%%%   This can be adjusted so that some of the turbines
%%%   can be fixed to a = 1/3.
numFixed = 0;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%   Turbine Parameters
turbineDiameter = 80;   % Diameter of the turbine [meters]
turbineSpacing = 7;     % Diameters
hubHeight = 70;         % Hub Height of the the turbine
                        % [meters]
roughness = 1e-4;       % surface roughness [meters]
                        % {from pg. 463 Erich Hau's
                        % Wind Turbines}

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%   Wind Parameters
rho = 1.225;            % [kg/m_3] Air Density
windSpeed = 9;         % [m/s] Speed of the wind

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%   Wake model Parameters
kappa = 0.5/log(hubHeight/roughness);  % Wake spreading
                                       % coefficient
                                       % [rise/run]

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%   Initialize the layout of the wind farm
%%%
%%%   This section can be modified to any 2D wind farm
%%%   layout desired so long as the format is perserved.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%   Layout for Horns Rev wind farm
numTurbines = 80;
space = turbineSpacing*turbineDiameter;
farmAngle = 48.585*pi/180;
x_offset = space - cos(farmAngle)*9.4*turbineDiameter;
y_offset = sin(farmAngle)*9.4*turbineDiameter;
turbineLayout = zeros(numTurbines,2);

for i = 1:10
    for j = 1:8
        k = (j-1)*10 + i;
```

```
        xVal = space*(i-1)-x_offset*(j-1);
        yVal = (j-1)*y_offset;
        turbineLayout(k,:) = [xVal yVal];
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%   Initialize the inputs for the Optimization Method
LB = zeros(1,numTurbines);
UB = 0.5.*ones(1,numTurbines);
initPop = (ones(numTurbines,1)./4)';

%%%   Wind directions to be evaluated
windDirVals = [221.415 270 312.173];

%%%   Initialize solution vectors
percentIncreaseVals = zeros(size(windDirVals));
plotVals = zeros(length(windDirVals),2);
plotVals(:,1) = windDirVals';

%%%   Loop through and evaluate each wind direction
for m = 1:length(windDirVals)
    tic
    windDir = windDirVals(m)

    %%%   Run the optimization of the axial
    %%%   induction factors
    [optimalAxialVals,fitVal] = patternsearch(...
        @findWindFarmFitness, ...
        initPop, [], [], [], [], LB, UB);

    %%%   Determine the optimal power produced
    optPower = -1*fitVal;

    %%%   Determine the regular power produced
    turbines = zeros(numTurbines,3);
    turbines(:,1:2) = turbineLayout;
    turbines(:,3) = (ones(numTurbines,1)./3)';
    otherVel = findInputVelocities(turbines);
    otherPower = findTotalPower(otherVel, ...
        (ones(numTurbines,1)./3)');

    %%%   Determine the percent increase in power
    percentIncreaseVal = 100*(optPower-otherPower)/...
        otherPower;
    percentIncreaseVals(m) = percentIncreaseVal;
```

```
%%%   Output Results
plotVals(:,2) = percentIncreaseVals'

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%   Post Processing

%%%   Create a grid of the optimized axial values
axialGrid = zeros(10,8);
for i = 1:10
    for j = 1:8
        k = (j-1)*10 + i;
        axialGrid(i,j) = optimalAxialVals(k);
    end
end
[X Y] = meshgrid(1:10,1:8);

%%%   Plot a contour map of the optimized values
figure(m);
hold on;
numLines = 7;
[mC hC] = contour(X, Y, axialGrid', numLines);
xlabel('Turbine Row Index','FontSize',14,...
    'FontWeight','bold');
ylabel('Turbine Column Index','FontSize',14,...
    'FontWeight','bold');
title(...
    {'Optimized Control for the Horns Rev Wind Farm',...
    ['with a ', num2str(windDir),'° Wind Direction']}...
    ,'FontSize',14,'FontWeight','bold');
plot(X,Y,'o','MarkerSize',5,'MarkerEdgeColor',...
    'k','MarkerFaceColor','k');
axis([0.5 10.5 0.5 8.5])
set(get(get(hC,'Annotation'),'LegendInformation'),...
    'IconDisplayStyle','Children');
%{
Assigns each line object's DisplayName property a string
based on the value of the contour interval it represents
%}
k =1;
ind = numLines;
hLines = get(hC,'Children');
while k < size(mC,2),
   set(hLines(ind),'DisplayName',num2str(mC(1,k)))
   k = k+mC(2,k)+1;
   ind = ind-1;
```

```
        end
        % Display the legend using DisplayName labels
        legend('show')

        toc
        disp('%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%');
    end

    %%%   Plot the performance increases vs wind direction
    figure(1);
    hold on;
    plot(windDirVals,percentIncreaseVals,'k');
    xlabel('Wind Direction [Degrees]','FontSize',14,...
        'FontWeight','bold');
    ylabel('Percent Power Increase','FontSize',14,...
        'FontWeight','bold');
    title('Optimization Increase vs. Wind Direction',...
        'FontSize',14,'FontWeight','bold');
    axis([215 315 0 1.1*max(percentIncreaseVals)]);
end
```

## B.4 findWindFarmFitness.m

```
function fitVal = findWindFarmFitness(axials)

    %%%   This function will take the axial induction factors
    %%%   from and individual in the sample population and
    %%%   compute the fitness.  The axial induction factors
    %%%   should be in a linear vector form.
    %%%
    %%%   The output fitness value will be the total power of
    %%%   the wind farm multiplied by negative one since a
    %%%   maximum power is desired but MATLAB optimizes to a
    %%%   minimum
    %%%
    %%%   Global Variables that need to be set:
    %%%
    %%%   turbineLayout:    A N by 2 matrix containing the x and
    %%%                     y locations of each turbine in the
    %%%                     wind farm
    %%%
    %%%   numFixed:         number of turbines not being
    %%%                     optimized but instead being fixed at
    %%%                     a = 1/3

    %%%   Constants used across different functions
```

```
    global turbineLayout numFixed


    numTurbines = length(axials) + numFixed;
    turbines = zeros(numTurbines,3);
    turbines(:,1:2) = turbineLayout;

    turbines(1:length(axials),3) = axials;
    turbines(length(axials)+1:numTurbines,3) = ...
        ones(numFixed,1)./3;

    velocities = findInputVelocities(turbines);
    totalPower = findTotalPower(velocities, turbines(:,3));
    fitVal = -1*totalPower;
end
```

### B.5 findTotalPower.m

```
function pTotal = findTotalPower(velocities, axials)

    %%%   This function will take the input velocities and axial
    %%%   induction factors and compute the power produced by
    %%%   the entire wind farm

    numTurbines = length(velocities);
    pSum = 0;
    for i = 1:numTurbines
        velocity = velocities(i);
        axial = axials(i);
        pCurr = findIndividualPower(velocity, axial);
        pSum = pSum + pCurr;
    end
    pTotal = pSum;
end
```

### B.6 findIndividualPower.m

```
function pInvidual = findIndividualPower(velocity, axial)

    %%%   This function will take the input velocity and
    %%%   axial induction factor and compute the power
    %%%   produced by the wind turbine

    %%%   Constants used across different functions
    global turbineDiameter rho
```

```
    C_p = 4*axial*((1-axial)^2);
    area = pi*(turbineDiameter^2)/4;
    pInvidual = (1/2)*rho*C_p*(velocity^3)*area;
end
```

### B.7 findInputVelocities.m

```
function result = findInputVelocities(turbines)

    %%%   This is a wrapper function that contains calls
    %%%   to all of the implemented wake models and
    %%%   determines which one should be called based on
    %%%   the global modelNumber variable.  If
    %%%   modelNumber isn't set the PARK model will be
    %%%   used.
    %%%
    %%%   modelNumber:
    %%%
    %%%   1 = PARK model
    %%%   2 = Mosaic Tile model

    %%%   Constants used across different functions
    global modelNumber

    if modelNumber == 2
        result = findMosaicTileInputVelocities(turbines);
    else
        result = findPARKInputVelocities(turbines);
    end
end
```

### B.8 findPARKInputVelocities.m

```
function result = findPARKInputVelocities(turbines)

    %%%   This function will go through and determine the input
    %%%   velocites for all of the turbines for a given turbine
    %%%   operational method using the PARK Wake Model.
    %%%   It should be noted that the following assumptions
    %%%   are made:
    %%%
    %%%     1.  Each turbine has the same diameter
    %%%     2.  Each turbine has the same hub height
    %%%     3.  Each turbine has the same surface roughness
    %%%
    %%%   The turbines input expects a N x 3 matrix where N is
```

```
%%%   the number of turbines.  The first column should
%%%   contain the x locations of each turbine.  The second
%%%   column should contain the y locations of each
%%%   turbine, and the third column should contain the
%%%   axial induction factor of the turbine.
%%%
%%%   tubines(i,:) = [x_i, y_i, a_i]
%%%
%%%   where i is the index of the turbine. i goes from 1 to
%%%   N
%%%
%%%   Other global values that need to be set prior:
%%%   kappa:            The wake speading coefficient
%%%   windSpeed:        The reference wind speed
%%%   turbineDiameter:  The diameter of the turbine
%%%   windDir:          The wind direction in degrees with
%%%                     0 = North = positive y-axis

%%%   Constants used across different functions
global kappa dependencyMatrix velocities theta windDir

%%%   Change from compass wind direction into cartesian
%%%   angle
theta = compass2angle(windDir)*pi/180;

%%%   Set up internal parameters
numTurbines = length(turbines(:,1));
velocities = -1.*ones(numTurbines,1);
dependencyMatrix = zeros(numTurbines,numTurbines);

%%%   Determine the wake spreading angle
alpha = atan(kappa);

%%%   Determine which turbines affect the performance of
%%%   other wind turbines
for i = 1:numTurbines
    x_i = turbines(i,1);
    y_i = turbines(i,2);
    for j = 1:numTurbines
        if i~=j
            x_j = turbines(j,1);
            y_j = turbines(j,2);
            beta = findBetaAngle(x_i,y_i,x_j,y_j);
            if beta < alpha
                %%%   Turbine i is in the wake of turbine j
                %%%   In other words turbine located in row
```

```
                    %%%   i depends on the turbine in column j
                    dependencyMatrix(i,j) = 1;
                end
            end
        end
    end

    for i = 1:numTurbines
        %%%  Only perform calculation if it has already not
        %%%   been done
        if (velocities(i) < 0)
            velocities(i) = findTurbineVelocity(turbines, i);
        end
    end

    result = velocities;
end

function velOut = findTurbineVelocity(turbines, index)

    %%%   This internal function implements the PARK
    %%%   Wake Model to determine the velocity
    %%%   deficit of a downstream wind turbine based on
    %%%   the number of wind turbines ahead of it

    global dependencyMatrix velocities windSpeed theta
    if sum(dependencyMatrix(index,:)) == 0
        velOut = windSpeed;
    else
        velSquareSum = 0;
        for j = 1:length(velocities)
            if dependencyMatrix(index,j) ~= 0
                %%%   "index" turbine is in the wake of the
                %%%   "j" turbine
                if velocities(j) < 0
                    %%%   "j" turbine input velocity has not
                    %%%   been computed
                    jIndex = j;
                    velocities(jIndex) = findTurbineVelocity(...
                        turbines, jIndex);
                end

                %%%   Determine the distance projected on
                %%%   the wind directions
                xDiff = turbines(index,1) - turbines(j,1);
                yDiff = turbines(index,2) - turbines(j,2);
```

```
                distance = abs(xDiff*cos(theta) + ...
                    yDiff*sin(theta));

                axial = turbines(j,3);
                vel_def = getVel_Def(axial, distance);
                velSquareSum = velSquareSum + vel_def^2;
            end
        end
        velOut = windSpeed*(1 - sqrt(velSquareSum));
    end
end

function deficit = getVel_Def(axial, distance)
    %%%   This function determines the velocity deficit
    %%%   based on the upstream axial induction factor
    %%%   and the distance between the turbines projected
    %%%   onto the wind direction
    global turbineDiameter kappa
    bVal = kappa/(turbineDiameter/2);
    gamma = 0.4; %%%   modified value to match Horns Rev
                 %%%   (default is 2)
    deficit = gamma*2*axial/((1 + bVal*distance)^2);
end

function beta = findBetaAngle(x_i, y_i, x_j, y_j)
    %%%   This function will determine if the turbine at
    %%%   (x_j,y_j) is within the wake of the turbine located
    %%%   at (x_i,y_i) for a given wind direction (theta).
    %%%   This caluclation can be found in Kusiak, A.
    %%%   "Design of wind farm layout for maximum wind
    %%%   energy capture".  Renewable Energy. Vol.  35.
    %%%   (2010) pg. 685-694

    global theta

    term1 = (x_j-x_i)*cos(theta);
    term2 = (y_j-y_i)*sin(theta);
    numerator = term1 + term2 + 1;

    denominator = sqrt( (x_j - x_i + cos(theta) )^2 ...
        + (y_j - y_i + sin(theta) )^2 );

    beta = acos(numerator/denominator);

end
```

```
function result = compass2angle(windDir)
    %%%   This function will take a given wind direction angle
    %%%   from a compass format and convert it into its
    %%%   corresponding cartesian form.  This assumes that
    %%%   the windDir angle is in degrees

    result = mod((360 - windDir + 90),360);
end
```

### B.9 findMosaicTileInputVelocities.m

```
function result = findMosaicTileInputVelocities(turbines)

    %%%   This function will go through and determine the input
    %%%   velocities for all of the turbines for a given turbine
    %%%   operational method using the Mosaic Tile Wake Model.
    %%%   It should be noted that the following assumptions are
    %%%   made:
    %%%
    %%%     1.  Each turbine has the same diameter
    %%%     2.  Each turbine has the same hub height
    %%%     3.  Each turbine has the same surface roughness
    %%%
    %%%   The turbines input expects a N x 3 matrix where N is
    %%%   the number of turbines.  The first column should
    %%%   contain the x locations of each turbine.  The second
    %%%   column should contain the y locations of each turbine,
    %%%   and the third column should contain the axial
    %%%   induction factor of the turbine.
    %%%
    %%%   tubines(i,:) = [x_i, y_i, a_i]
    %%%
    %%%   where i is the index of the turbine. i goes from 1 to
    %%%   N
    %%%
    %%%   Other global values that need to be set prior:
    %%%   kappa:            The wake speading coefficient
    %%%   windSpeed:        The reference wind speed
    %%%   turbineDiameter:  The diameter of the turbine
    %%%   windDir:          The wind direction in degrees with
    %%%                     0 = North = positive y-axis

    %%%   Constants used across different functions
    global kappa dependencyMatrix velocities theta windDir

    %%%   Change from compass angle to cartesian angle
```

```
    theta = compass2angle(windDir)*pi/180;

    %%%   Initialize internal Parameter
    numTurbines = length(turbines(:,1));
    velocities = -1.*ones(numTurbines,1);
    dependencyMatrix = zeros(numTurbines,numTurbines);

    %%%   Determine wake spreading angle
    alpha = atan(kappa);

    %%%   Determine which turbines affect the performance
    %%%   of other wind turbines
    for i = 1:numTurbines
        x_i = turbines(i,1);
        y_i = turbines(i,2);
        for j = 1:numTurbines
            if i~=j
                x_j = turbines(j,1);
                y_j = turbines(j,2);
                beta = findBetaAngle(x_i,y_i,x_j,y_j);
                if beta < alpha
                    %%%   Turbine i is in the wake of turbine j
                    %%%   In other words turbine located in row
                    %%%   i depends on the turbine in column j
                    dependencyMatrix(i,j) = 1;
                end
            end
        end
    end

    %%%   Evaluate the velocities of all the wind turbines
    for i = 1:numTurbines
        %%%  Only perform calculation if it has already not been
        %%%   done
        if (velocities(i) < 0)
            velocities(i) = findTurbineVelocity(turbines, i);
        end
    end

    result = velocities;
end

function velOut = findTurbineVelocity(turbines, index)

    %%%   This internal function implements the Mosaic
    %%%   Tile Wake Model to determine the velocity
```

```
%%%   deficit of a downstream wind turbine based on
%%%   the number of wind turbines ahead of it

global dependencyMatrix velocities windSpeed theta
if sum(dependencyMatrix(index,:)) == 0
    %%%   turbine is in front
    velOut = windSpeed;
elseif sum(dependencyMatrix(index,:)) == 1
    %%%   only one turbine ahead
    notDone = 1;
    j = 1;
    while notDone == 1
        if dependencyMatrix(index,j) ~= 0
            %%%   "index" turbine is in the wake of the "j"
            %%%   turbine
            if velocities(j) < 0
                %%%   "j" turbine input velocity has not
                %%%   been computed
                jIndex = j;
                velocities(jIndex) = ...
                     findTurbineVelocity(turbines, jIndex);
            end

            %%%   Determine the distance projected
            %%%   onto the wind direction
            xDiff = turbines(index,1) - turbines(j,1);
            yDiff = turbines(index,2) - turbines(j,2);
            distance = abs(xDiff*cos(theta) + ...
                yDiff*sin(theta));

            axial = turbines(j,3);
            vel_def = getVel_Def(axial, distance, 1);
            notDone = 0;
            velOut = velocities(j)*(1-vel_def);
        end
        j = j+1;
    end
else
    %%%   multiple turbines ahead

    %%%   Determine the number of turbines ahead
    numTurbAhead = sum(dependencyMatrix(index,:));

    %%%   First need to find the closest turbine
    minDist = Inf;
    minJ = -1;
```

```
        for j = 1:length(velocities)
            if dependencyMatrix(index,j) ~= 0
                %%%   "index" turbine is in the wake of the "j"
                %%%   turbine
                xDiff = turbines(index,1) - turbines(j,1);
                yDiff = turbines(index,2) - turbines(j,2);
                dist = abs(xDiff*cos(theta) + yDiff*sin(theta));
                if dist < minDist
                    minDist = dist;
                    minJ = j;
                end
            end
        end

        %%%   minJ is the index of the closest turbine ahead
        if velocities(minJ) < 0
            %%%   "j" turbine input velocity has not been
            %%%   computed
            velocities(minJ) = findTurbineVelocity(...
                turbines, minJ);
        end
        axial = turbines(minJ,3);
        vel_def = getVel_Def(axial, minDist, numTurbAhead);
        velOut = velocities(minJ)*(1-vel_def);
    end
end

function deficit = getVel_Def(axial, distance, numAhead)

    %%%   This function computes the velocity
    %%%   deficit given the axial induction factor,
    %%%   the downstream distance, and the
    %%%   number of turbines ahead of the
    %%%   downstream wind turbine
    global turbineDiameter
    D_o = turbineDiameter;
    gamma = 0.8;
    if numAhead == 1
        k = 3;
        alpha = 1.2;
    else
        k = 2;
        alpha = 0.7;
    end
    beta = (1/2)*(2-axial)/(1-axial);
    betaTerm = beta^(k/2);
```

```
    alphaTerm = alpha*distance/D_o;
    xi = max(betaTerm,alphaTerm);
    psi = gamma*numAhead;
    deficit = axial/( ( xi^(2/k) )*(psi^2) );
end

function beta = findBetaAngle(x_i, y_i, x_j, y_j)
    %%%   This function will determine if the turbine
    %%%   at (x_j,y_j) is within the wake of the turbine
    %%%   located at (x_i,y_i) for a given wind direction
    %%%   (theta).  This caluclation can be found in
    %%%   Kusiak, A.  "Design of wind farm layout for
    %%%   maximum wind energy capture".  Renewable Energy.
    %%%   Vol.  35.  (2010) pg. 685-694

    global theta

    term1 = (x_j-x_i)*cos(theta);
    term2 = (y_j-y_i)*sin(theta);
    numerator = term1 + term2 + 1;

    denominator = sqrt( (x_j - x_i + cos(theta) )^2 ...
        + (y_j - y_i + sin(theta) )^2 );

    beta = acos(numerator/denominator);

end

function result = compass2angle(windDir)
    %%%   This function will take a given wind
    %%%   direction angle from a compass format
    %%%   and convert it into its corresponding
    %%%   cartesian form.  This assumes that
    %%%   the windDir angle is in degrees

    result = mod((360 - windDir + 90),360);
end
```