Engineering an Optimal Wind Farm

Department of Wind Energy Master Report

Stjepan Mahulja

DTU Wind Energy Master Thesis

11/09/2015

DTU Vindenergi
Institut for Vindenergi

Title: Engineering an Optimal Wind Farm

Author: Stjepan Mahulja
Department: European Wind Energy Master - EWEM

Summary (max. 2000 characters):
The project covers creation of a framework for optimisation of wind farm layouts. The optimisation is done with an objective of maximising the Net Present Value of the farm. With that aim, both power production and the fatigue driven component degradation are considered.
The work is split in two main parts. First, in order to circumvent the use of full DWM model simulations, surrogate models of lifetime equivalent loads and power production are created. In the second stage, an optimisation platform is assembled as a two-stage, multi-fidelity strategy encompassing genetic algorithm for global and gradient based algorithm for local optimisation.
The analysis of a wind farm is performed using DWM surrogate model which significantly contributes to the speed of computations. On the other hand, the use of surrogates makes it possible to capture the in-stationary effects of wake meandering with sufficient quality.
The objective of the thesis is to show the steps required in assembling a framework such as this. Moreover, through the tests performed on Middelgrunden wind farm, the aim is to show the importance of setting the number of turbines as a design variable.

11/09/2015

Project Period:
13.04.2015. - 11.09.2015.

ECTS:
30

Education:
Master of Science

Field:
Wind Energy

Supervisors:
Gunner Chr. Larsen (DTU)
Ali Elham (TU Delft)

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

Contract no.:

Project no.:

Sponsorship:

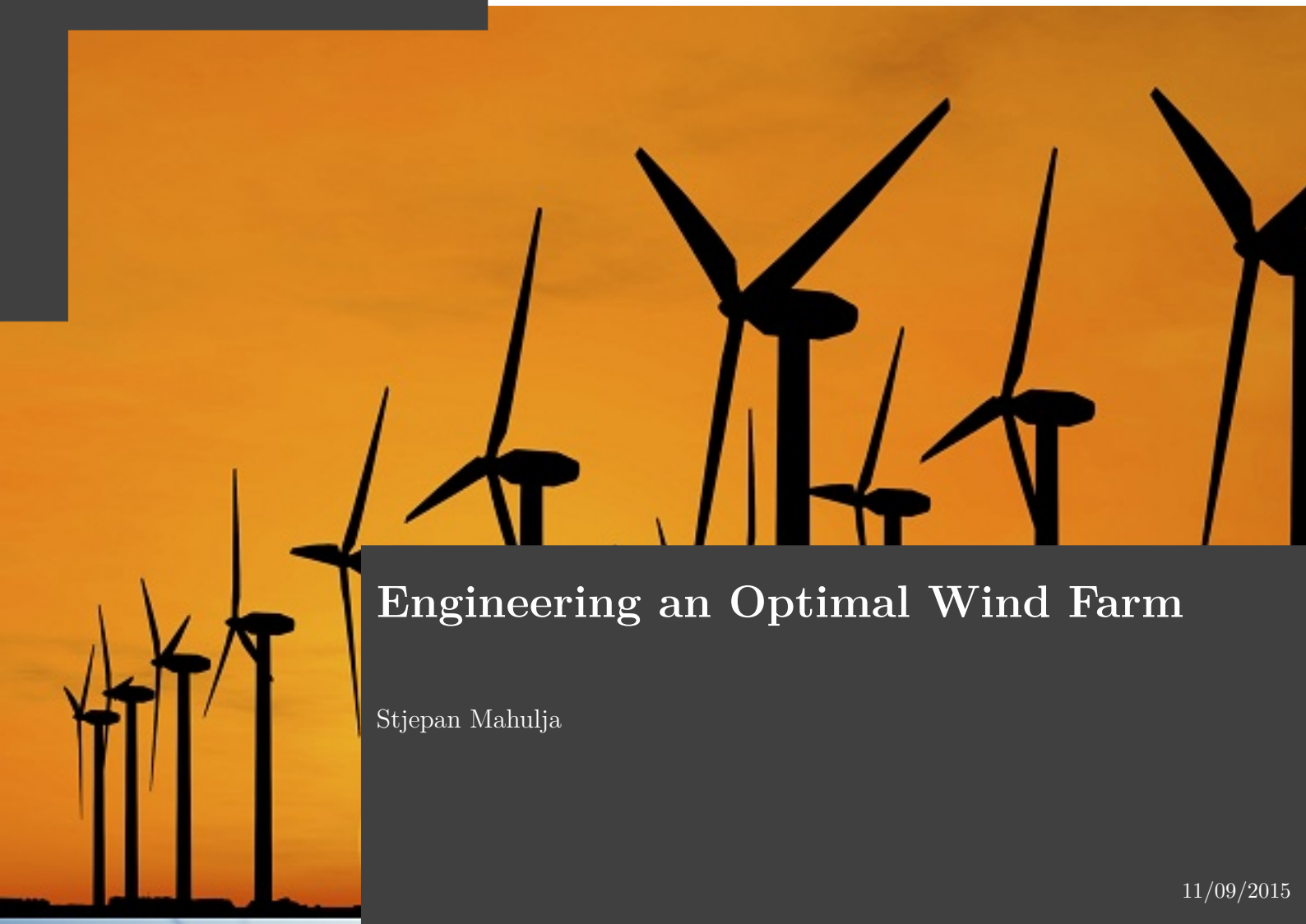
Front page:
Cover_DTU_Vind

Pages: 174
References: 43

Danmarks Tekniske Universitet
DTU Vindenergi
Nils Koppels Allé
Bygning 403
2800 Kgs. Lyngby

www.vindenergi.dtu.dk

Master Thesis of Science



Engineering an Optimal Wind Farm

Stjepan Mahulja

11/09/2015

European Wind Energy Master - EWEM- Rotor Design Track

Engineering an Optimal Wind Farm

Master of Science Thesis

For obtaining the degree of Master of Science in Engineering Wind Energy at Technical University of Denmark and in Aerospace Engineering at Delft University of Technology.

Stjepan Mahulja

11/09/2015

European Wind Energy Master - EWEM
DUWIND - Delft University of Technology
Risø - Denmark Technical University

European Wind Energy Master - EWEM
Of
Rotor Design Track

The undersigned hereby certify that they have read and recommend to the European Wind Energy Master - EWEM for acceptance a thesis entitled **"Engineering an Optimal Wind Farm"** by **Stjepan Mahulja** in partial fulfillment of the requirements for the degree of **Master of Science**.

Dated: 11/09/2015

Supervisor: _____
Gunner Chr. Larsen of DTU

Supervisor: _____
Ali Elham of TU DELFT

Reader: _____
REVIEWER of

Declaration

I, Stjepan Mahulja, declare that this thesis titled, "Engineering an Optimal Wind Farm" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: 

Acknowledgements

I would like to express my very great appreciation to Gunner Larsen whose extensive coverage of the topic is so vast that it was hard to find anything new to do.

I would also like to offer my special thanks to Ali Elham who had great advices in crucial moments.

My thanks are extended to David Robert Verelst and Torben J. Larsen for the help with Gorm.

I would also like to than the initiators and coordinators of EWEM and the complete staff of both DTU and TU Delft for making my stay throughout these two years very comfortable.

Finally, I would like to thank all my friends and colleagues for being kind enough to go for summer holidays and let me finish this project in piece.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Following report is a result of a master thesis project carried out at Technical University of Denmark and Delft University of Technology. The author presents this project as the final work of the double degree master programme named European Wind Energy Master. Although the author took the specialisation in the aerodynamic design of wind turbine rotors, something related to wind farms and optimisation algorithms was just too attractive to miss out. Therefore, the scope of this work will cover optimisation of wind farm layouts. The literature study, and exploration of the wind farm layout optimisation as a topic, has been covered, and documented in the preceding report.

The structure of this project is given as follows. First, a short introduction in the topic of wind energy and some present trends is given. The discussion continues towards wind farms and their role in producing clean efficient energy. For the first time in the project, the problem of planning a wind park layout is mentioned. Definition of the problem this project will take on is followed by the analysis of methods and tools required to successfully accomplish the goal which is "How to engineer an optimal wind farm layout?".
Last section of the introduction gives a short view of some parameters which are commonly used to describe a wind farm. This serves to get a better understanding of what stands behind values associated with particular wind farms.
Next chapter present the surrogate modeling as a way of reducing the computational effort during evaluation of a wind farm. Subsequently, methodologies for building a surrogate model are described, and surrogate models of several wind turbine outputs is presented.
Following chapter presents two possible algorithms applied for the purpose of designing the internal grid of a wind farm. These tools have the goal to find the minimal cable length for a given layout.
Fourth chapter presents a framework for optimising wind farm layouts. A platform

encompassing two-stage optimisation is built. In the first stage, a genetic algorithm is used on a discretized area. Second stage uses the gradient search algorithm to further tune the optimal layout by shitting the turbines in a continuous domain. The analysis of the wind farm performance is performed using the developed surrogate model.
Finally, a test has been carried on Middelgrunden wind farm, where the layout was optimised for the objective of maximising the net present value of the farm.

1.1 Wind energy

Wind is a phenomena caused by the uneven heating of the Earth's surface by the sun. Due to the difference in the atmospheric pressure, the wind manifests as an air flow originating in the high pressure and moving towards low pressure area. Significant influence to the large scale movement of such air patterns is also due to the Coriolis force which appears as a consequence of Earth's rotation.
We perceive the wind by its kinetic energy, i.e. as a speeding flow that carries the leaves and occasionally an umbrella. In fact, we feel the wind as a pressure on our body that pushes us with an invisible force.
The wind is a flow of energy. Though it my seem fragile because it changes its form form kinetic to potential, or even to heat, very quickly, wind hold enormous power. Being an energy flow rather than an energy container, it is defined as a renewable source of energy, which means is is naturally replenished on a human timescale.
Wind energy is harvested by wind turbines which convert it into electrical energy. This is done using the blades whose aerodynamic shape generates the difference in pressure which results in a force. The force is applied on a rotor which turns the generator shaft and the generator then converts mechanical energy to electricity. The conversion of wind into electrical energy does not produce any harmful emissions making the wind a clean and sustainable source of energy.
Producing large amounts of energy is done by clustering many wind turbines together in so called wind farms or wind parks. These represent a wind-fuelled version of a traditional power plant and will be the main focus of this work.

Over the years, wind energy has grown into a serious industry and is now competitive with the fossil fuel based energy technologies. Globally, the accumulated installed wind power capacity has increased eight times in the past ten years (IRENA report Renewable Power Generation Costs in 2014). The estimates predict the wind to be the leading renewable energy source and to provide up to 34% of the electricity in EU by 2030 (The European

Technology Platform for Wind Energy TPWind). To achieve this, the primary driver is reducing the cost of energy (CoE) in the development of new technologies, but also to improve the predictability and system services related to production of wind energy and its higher penetration into the grid and market.
Due to larger available areas, less impact to the environment and most of all higher wind resource, the largest growth is expected in the offshore sector. However, due to the requirements regarding the site visits, special equipment and downtime due to the maintenance, the offshore wind power is still more expensive than its onshore counterpart. As a consequence, one of the parameters recently attracting more attention are the O&M costs which should have a significant impact to the reduction of the costs. A further space for improvement is available in reducing the much higher capital expenditure of cabling and support structures [24].

Final determining effect on the electricity price is made based on the operation of the turbines within the farm. If planned properly, wind farm will benefit from the technology implemented in the turbines. On the other hand, if some aspects are neglected during the farm development, the performance of the farm may be poor, no matter how advance the turbines are. It is thus necessary to analyse wind farms on a system level and identify aspects through which they may be improved. As it will turn out, smart wind farm layout is able to reduce the O&M costs and balance the investment w.r.t. the revenue which will result in the greatest value of the farm.

1.2 Wind farm layout optimisation problem (WFLOP)

Wind farms or wind parks are power plants which consist of a number of wind turbines positioned over large area of land or water. The goal of a wind farm is to harness large amounts of kinetic energy from the wind and convert it into electricity, but also to do it in an efficient way. This project focuses around the efficiency and ways how it can be improved. Two design variables are used to achieve this goal, these are the **number of wind turbines** and their **position within the wind farm**. Apart from the wind turbine type, these two parameters are the only variables that may be used to obtain a better performance of a wind park. Because these two variables define the layout of the park, literature defines this problem as the Wind Farm Layout Optimisation Problem (WFLOP).
Michele Samorani describes the WFLOP as:
".. optimally positioning the turbines within the wind farm so that the wake effects are minimized and therefore the expected power production maximized" [39].

It is true the wake effects should be minimised, however, in order to optimise not only the farm's performance, but also its feasibility, considering power production alone is not enough. Logic reasons the optimisation of layouts has to take into account the value which defines the wind farm on the financial market. Gunner Larsen explains:
"...to achieve the optimal economic output from a wind farm over its lifetime, an optimal balance between capital costs, operation and maintenance (O&M) costs, fatigue lifetime consumption of turbine components and power production output is to be determined on a rational background" [23].
To eliminate bias towards any particular solution, the optimisation needs to operate freely, i.e. is must to be allowed to find even the most unlikely layout if necessary. Consequently, engineering an optimal wind farm layout is carried out by determining both the number and the position of wind turbines while targeting the objective of profitable production of energy.

1.2.1 Objective function

By mathematical formulation, optimisation is defined as a minimisation of the function. However, at this stage, without going into details regarding the formulation, it is more convenient to think of the objective as the financial balance (see Equation 1.1). In that sense, the objective is to maximise the difference between the profit (P) and the costs (C). Design variables used to this end are the number of turbines (N), which is an integer variable, and the coordinates ($\mathbf{X} = (x, y)$) of each turbine. Note however the following:

- In the genetic algorithm, the coordinates are pre-defined on a grid. Thus, the variable $\mathbf{X}$ reduces to the choice of a grid point.

- In the gradient search algorithm, it is more appropriate to use the offset from the starting point as a design variable rather than the coordinate itself.

$$\max_{N \in \mathbb{Z}^+, \mathbf{X} \in \Re^{2N}} \quad f(N, \mathbf{X}) = P - C \tag{1.1}$$

To further proceed with the discussion on how the objective function is computed, it is useful to understand on which parameters the profit and costs depend. With this aim, next section gives an overview of wind farm analysis.

1.3 Wind farm analysis

Following section reflects on the steps commonly taken to sufficiently describe a wind farm. Such a description serves the purpose of summarising important parameters which characterise the wind farm by its energy production, efficiency, lifetime expectancy and economic feasibility. Consequently, a non-biased approach of evaluating and comparing wind farms is presented with the aim of proposing steps for improving of their layout.

1.3.1 Site Conditions

Simulation of the wind farm in operation first requires a model of the site conditions where the wind farm is located. Such a model is built from an analysis of long term measurements taken at the site. Subsequently, the model provides environment for simulating each wind turbine in the layout. These results, representing a wind farm on a component level, are then summed or averaged to describe the farm on a system level.
The site conditions are modeled through:

- **Wind distribution in terms of wind speed and wind direction**
  These are commonly given as 2-parameter Weibull distributions of wind speeds for 12 wind directions (sectors) together with the probability of wind coming from each of these (i.e. the site wind distribution conditioned on the wind direction and the wind direction distribution).
- **Turbulence** [1]
  - Turbulence intensity at the site
    While it is possible to analyse the turbulence parameters from the measured data and model the conditions at the site using these, in this project the turbulence is modeled using a reference turbulence intensity I_{ref} given by the IEC standard [1]. As such, the sites are assigned in three classes as shown in Figure 1.1. "A" designates the site characterised by high turbulence intensity, "B" by medium turbulence intensity and "C" designates the site of low turbulence intensity. The turbulence intensity, used as input to HAWC2, is given as Equation 1.2 for normal turbulence model (NMT) and is related to I_{ref} as shown in Equation 1.3. Note that "model" here does not refer to turbulence modeling. Most often, as

[1] The term "turbulence" is used to describe fluctuation from the mean wind velocity and is given for each of the three velocity components.

in this project, the turbulence is modeled by the Mann model [27]. In such a case the turbulence is characterised as isotropic assuming neutral atmospheric conditions.

$$TI = \frac{\sigma}{\bar{V}} \tag{1.2}$$

where the standard deviation is given as

$$\sigma = I_{ref} \cdot 0.75 \cdot V + 5.6 \tag{1.3}$$

The actual turbulence intensity at each turbine is altered by the presence of the turbine itself and the upstream emitted wake. This is accounted for by the DWM model [21].

– Length scale of turbulence

Length scale of turbulence indicates size of representative vortices loading the wind turbine. DWM model uses 3 turbulent boxes.

The standard Mann turbulent box is characterised by the turbulent length scale $L = 0.7\Lambda$, where the wave length is $\Lambda = 42$ for hub heights above 42m. The recommended shear blocking factor for normal atmospheric conditions is $\Gamma = 3.9$.

Apart from the standard Mann turbulent box, two additional turbulent boxes are used. The first one describing the meandering process is characterised by the same turbulent length scale. However, it covers a larger volume with a coarser grid and includes grid-volume averaging of the turbulence to isolate the large turbulence scales dictating the wake meandering.

Third turbulence box follows meandering deficit and accounts for the wake self generated small scale turbulence. The wave length of these are in the order of $1D \cdot \frac{1}{16}$.

For more details regarding the settings of each box, reader is pointed to [25].

Wind turbine class		I	II	III	S
V_{ref}	(m/s)	50	42,5	37,5	Values
A	I_{ref} (-)	0,16			specified
B	I_{ref} (-)	0,14			by the
C	I_{ref} (-)	0,12			designer

Figure 1.1: *Basic parameters for wind turbine classes. V_{ref} is the reference wind speed average over $10min$ and I_{ref} is the expected mean value of the turbulence intensity at $15m/s$. Reproduced from [1]*

- **Wind sheer profile**

  The wind sheer profile is chosen based on the surface roughness conditions. The wind sheer describes the variation of the mean wind speed with the distance from the ground/sea. Common model are the linear profile given in Equation 1.4, logarithmic profile given in Equation 1.5 or the power law profile given in Equation 1.6. The profiles are compared in Figure 1.2.

$$\bar{u}(z) = \Delta u \cdot z \tag{1.4}$$

  where

  Δu is the con stand gradient of wind speed with height

  z is the height from the ground

$$\bar{u}(z) = u_0 \frac{\log(\frac{z}{r_0})}{\log(\frac{z_0}{r_0})} \tag{1.5}$$

  where

  z_0 is the reference height used for fitting the profile

  z is the height from the ground

  r_0 is the roughness length

$$\bar{u}(z) = u_0 \left(\frac{z}{z_0}\right)^{\alpha} \tag{1.6}$$

  where

  z is the height from the ground

  α is the wind sheer or power law exponent

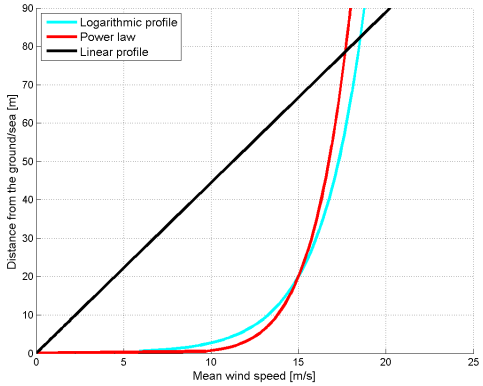


Figure 1.2: *Comparison of wind sheer profiles*

- **Sea, elevation and obstacles**

  The sea conditions, apart from the water depth, are not considered in this project. Furthermore, the water depth was considered only for the purpose of estimating the cost of the support structure. Future work on this topic may consider the wave effect as an additional factor influencing the component fatigue degradation. Although the author is not especially familiar well with the topic of waves and its effect to the loading, an assumption is that the wave conditions will not vary considerably over the wind farm. Thus, their effect should not influence the optimal layout.

  The elevation may be accounted for as a factor in the cost estimation model. However, because the target of this project are large wind farms, especially those offshore, it is not likely the site will be characterised by abrupt changes in elevation. Thus, elevation is not taken into account. Same reasoning is applied to obstacles.

  When the site is located near shore, depending on the wind direction, the site conditions may be modeled by different reference turbulence intensity and different wind shear profile (i.e. low turbulence intensity when wind comes from sea and medium turbulence intensity when wind comes from land). This was not implemented in the project.

Parameters that have been mentioned are necessary to create the site model. Once this is done, the simulation environment is ready for the input of wind turbine. In this project, the turbine used in all studies is the NREL 5MW wind turbine [15]. No further discussion about it is made in the rest of the work. Moreover, most of the simulations results are non-dimensionalized with the aim of making observations and conclusions valid in general. Exception is made for the estimation of economic feasibility. Simulation of wind turbines in the site specific conditions is done using the code supplied along with the report. In

order to reduce the computational effort, the code uses surrogate models created using HAWC2 and the DWM model as explained in section 2.1.

1.3.2 Describing a wind farm

A wind farm is a system, however, a detailed analysis of each of its components, in this case wind turbines, is crucial to understand and describe its performance.
A wind farm is characterised by the following:

- **The layout**
  Which is described by:
  - **Area available for positioning the turbines**
    Considering the layout, area is defined only by its boundaries. In general, properties such as elevation/depth and type of soil play a role in wind farm planning.
  - **Number of wind turbines in the farm**
  - **Location of the wind turbines within the area**
    Positioning of the turbines will as a consequence determine the spacing between the turbines. The spacing is an important parameter since it gives an estimation of the wake effect.
- **Type of wind turbines in the farm**
  The important parameters are the rated power, rotor diameter, hub height and various costs. Based on the type of turbine used, following inputs to the wind farm model are required to perform simulations of the park:
  - **The power output**
    Strictly speaking, power output of the wind turbine is a signal varying in time. However, from now on the term "power output" will refer to a value of power output averaged over time and given as a discrete value depending on wind speed, spacing and the wake angle.
  - **The loading**
    Loading is usually computed through aeroelastic analysis for given conditions. Example of a tool used for this is HAWC2. However, considering the computational effort, it is not practical to perform aeroelastic simulation of all individual turbines in the farm for every situation. The loading is obtained

as a time-series signal, and the fatigue driven degradation it causes depends on the history of the signal. Thus, it is impractical to store the loading as a discrete value. For this reason, the measure of lifetime equivalent load is used.

- **Rated capacity**
  The sum of rated powers of each wind turbine in the farm.

- **Energy production**
  Energy production is usually given as aggregated value referring to a period of one year (Annual Energy Production - AEP) and is computed as shown in Equation 1.7. Each wind turbine is considered in the centre of a polar coordinate system which is divided in 12 sectors, each characterised by one wind direction (from 0° to 330°).

$$AEP = 8760 \sum_{i=1}^{N} \sum_{j=1}^{12} \left(\sum \left(P(\lambda_{i,j}, \alpha_{i,j}) \cdot f(k_j, A_j) \right) \cdot p_j \right) \tag{1.7}$$

$\sum_j$ is the sum over 12 sectors

$\sum_{i=1}^{N}$ is the sum over N wind turbines

$P(\lambda_{i,j}, \alpha_{i,j})$ is the vector of power outputs for each wind speed bin. The power output is a function of spacing λ and the wake angle α.

The spacing λ and the wake angle α are invariant w.r.t. wind speed, but vary for each wind turbine over each sector.

$\mathbf{f}(k, A)$ is the vector of wind speed bin probabilities computed from cumulative Weibull distribution as shown in Equation 1.8.

k_j is the Weibull shape parameter varying over each sector.

A_j is the Weibull scale parameter varying over each sector.

p_j is the probability of a wind coming from sector j

8760 is the number of hours in a year

In order to speed up the computations, AEP is computed as a matrix product. Example showing the mapping is given for power in Equation 1.9, probability of wind speed bin in Equation 1.10, and sector probability in Equation 1.11. Rows of power matrix (Equation 1.9) contain power outputs for each wind speed bin. Column of power matrix is power output characteristic for sector one sector. Each turbine is described by 12 columns. Note again that $P_{k,j}$ in Equation 1.9 is a function of λ and α. Wind speed bin probability matrix (Equation 1.10) contains probability of each wind speed bin in each row. First 12 columns are unique (i.e. describe 12 sectors) and repeat for each wind turbine. The wind direction probability is invariant with wind speed, thus, the matrix (Equation 1.11) has identical rows. Moreover, only values in the first 12 columns (12 sectors) are unique, after which they are repeated for each wind turbine (every 12 columns).

$$\mathbf{f}(k, A) = \begin{bmatrix} f(x_1, x_2; k, A) \\ \vdots \\ f(x_{22}, x_{23}; k, A) \end{bmatrix} \tag{1.8}$$

$$f(x_1, x_2; k, A) = -e^{-(\frac{x_2}{A})^k} + e^{-(\frac{x_1}{A})^k}$$

where x is the vector containing wind speeds from cut-in to cut-out (23 values from 4m/s to 26m/s).

$$P = \left[\begin{bmatrix} P_{k=1,j=1} & \dots & P_{1,12} \\ \vdots & & \vdots \\ P_{22,1} & \dots & P_{22,12} \end{bmatrix}_{i=1} , \dots , \begin{bmatrix} P_{1,1} & \dots & P_{1,12} \\ \vdots & & \vdots \\ P_{22,1} & \dots & P_{22,12} \end{bmatrix}_{N} \right] \tag{1.9}$$

$$f = \left[\begin{bmatrix} f_{k=1,j=1} & \dots & f_{1,12} \\ \vdots & & \vdots \\ f_{22,1} & \dots & f_{22,12} \end{bmatrix} , \dots , \begin{bmatrix} f_{1,1} & \dots & f_{1,12} \\ \vdots & & \vdots \\ f_{22,1} & \dots & f_{22,12} \end{bmatrix} \right] \tag{1.10}$$

$$p = \left[\begin{bmatrix} p_{j=1} & \dots & p_{12} \\ \vdots & & \vdots \\ p_1 & \dots & p_{12} \end{bmatrix} , \dots , \begin{bmatrix} p_1 & \dots & p_{12} \\ \vdots & & \vdots \\ p_1 & \dots & p_{12} \end{bmatrix} \right] \tag{1.11}$$

In Equation 1.9 to Equation 1.11 the wind speed bins are denoted by index k, sectors by index j, and wind turbines by index i.

- **Capacity factor**
  Capacity factor is defined as AEP over the annual production at the full capacity (see Equation 1.12).

$$CF = \frac{AEP}{8760 \cdot CAP_{rtd}} \tag{1.12}$$

  where $CAP_{rtd} = \sum_{i=1}^{N} P_{rtd}$ is the rated capacity

- **Efficiency**
  The efficiency η of a wind farm is given as the ration between the annual energy production computed as in Equation 1.7 and the annual energy production computed using the same equation, but instead of reading the power from the surrogate (as function of spacing and wake angle), "clean" power curve is used. Thus, the efficiency accounts for the losses due to the wakes. The wake loss is given as $1 - \eta$.

- **Fatigue damage**
  Degradation of the structural member due to fatigue [2] is computed as the ratio of an equivalent load accumulated over a period of 20 years (termed the lifetime equivalent load), and the maximal equivalent load the turbine component can withstand. Given as the percentage, fatigue damage estimates how much of the turbine component has been used throughout the lifetime. The equivalent loads are measured at various locations of different components of the turbine. These are for example moments at the blade root, tower base and the shaft. (These are critical components)
  The estimation of maximal equivalent load the turbine can withstand requires detailed knowledge about the material used for the specific part of the turbine, high fidelity simulations and/or experimental measurements. As a consequence, the true value for the maximum equivalent load is often not available. Nevertheless,

[2] Fatigue is defined as a stress caused by the oscillating loading exerted on a structure. Fatigue stress is measured by counting the cycles in the loading signal and sorting them in different bins. Using these, an equivalent load is computed for specified number of cycles characteristic for the structure in question. For example, the blades of a wind turbine are considered to experience number of cycles in the order of 10^6. The equivalent load is thus the load structure would experience, if it was subjected to some characteristic number of cycles. The important fact is that the damage caused by the equivalent load is equivalent to the damage, which occurs by exerting original loading on the structure. The processing of the loading signal and the subsequent computation of the equivalent load is done using the rain-flow counting algorithm combined with Palmgren-Miner fatigue accumulation rule.

damage due to fatigue degradation may be shown w.r.t. equivalent loads other than the maximum one. Although the percentage will not give a correct estimation on the absolute scale, its representation as relative change in value when subjected to optimisation will be correct. The challenge is, however, in determining the weighting of fatigue driven degradation costs compared to the positive cash flow from power production.
Fatigue driven degradation of components is will be expressed through factor shown in Equation 1.13. This factor will be used to scale the costs of O&M.

$$f_{FD} = w_{comp} \frac{M_{eq}}{M_{max,eq}} \tag{1.13}$$

where M_{eq} stands for equivalent moment, and w_{comp} is the weight reflecting the relative price of each component.

- **Costs**
  Wind farm description cannot be complete without mentioning or at least indicating some level of costs. The feasibility of a wind farm and subsequent decision to build it depends on whether the wind farm is able to generate revenue. An estimation of the profit encompasses various deductions and expenses of which the notable ones are mentioned here.

  – **The investment cost**
    Planning and developing a wind farm is a complex, long-term process the costs of which can hardly be thoroughly described by someone lacking extensive experience in the field. An example of costs for offshore wind farms is given in Figure 1.4. Out of these, the following costs vary w.r.t. the parameters mentioned so far:

    * **Cost of turbines**
      According to [26] price of a wind turbine may be estimated using Equation 1.14. Compared to Figure 1.4, Equation 1.14 overestimates the costs a bit.

      $$C_{WT} = -0.15 + 0.92 P_{rtd} \tag{1.14}$$

      where P_{rtd} is given in [MW] and C_{WT} in [M€]

    * **Cost of cabling**
      Cost estimations for cabling are given in Figure 1.3. Design of electric grid for the wind farm is covered in chapter 3 (section dealing with determining

shortest cabling length).

In € / m	Offshore	Onshore
Cable	270	135
Installation	405	135
Total	675	270

Figure 1.3: *Assumed prices for cable cost per running meter. Reproduced from [36]*

∗ **Cost of support structures**

The cost of foundations for offshore wind farms is estimated to be proportional to the water depth as shown in Equation 1.15 [36].

$$C_F = 0.2C_{WT} + 0.02C_{WT}(WD - 8) \tag{1.15}$$

where WD is the water depth in meters at the site of the turbine

	Investments (€1000 /MW)	Share (%)
Turbines ex-works, including transport and erection	815	49
Transformer station and main cable to coast	270	16
Internal grid between turbines	85	5
Foundations	350	21
Design and project management	100	6
Environmental analysis	50	3
Miscellaneous	10	<1
Total	1680	~100

Note: Exchange rate EUR1 = DKK7.45.
Source: Risø DTU

Figure 1.4: *Average investment costs per MW related to offshore wind farms in Horns Rev and Nysted. Reproduced from [30]. Horns Rev consists of 80 2MW turbines, Nysted consists of 72 2.3MW turbines.*

– **Maintenance costs - costs due to fatigue degradation**

Minimising O&M is done by including the cost of component degradation into the objective function guiding the optimisation. Modeling costs due to fatigue degradation is reasoned as follows. Increased wear of components requires more frequent maintenance, the cost of which may be considered as a percentage of wind turbine price or proportional to the investment.

Maintenance cost are estimated to account in the order of 20% to 25% of the total levelized cost per kWh produced over the lifetime of the turbine [30]. In

this project three turbine components are considered for fatigue damage. Cost of blades, tower and the shaft have a relation of $0.84 - 1 - 0.08$. Thus, the cost of maintanance due to fatigue degradation is estimated as a percentage of the LCoE (Equation 1.19) and scaled using fatigue damage factor shown in Equation 1.13. This result in Equation 1.16.

$$C_{O\&M} = 0.25 C_I \sum_i f_{FD,i} \tag{1.16}$$

where the investment is given as in Equation 1.17

$$C_I = \frac{N_{turb} \cdot C_{WT} + C_{cab}}{0.4} \tag{1.17}$$

where $C_{cab} = p_{cab} \cdot L_{cab}$ with the cable price p_{cab} taken from Figure 1.3 and length of cable computed by algortihm described in chapter 3.
Factor 0.4 is used to account for the remaining cost of developing the wind farm.

– **Cost of money**
Large projects, such as development of wind farms, require funding from various financial institutions. While detailed explanation of funding is not within the scope of this project, one may simplify the situation by accepting the fact that a sum of money has to be borrowed, and a bit larger sum of money has to be returned. One should also accept the fact that the sum of money today and the same sum tomorrow are not valid equally. Cost of money C_e may be expressed as in Equation 1.18 [20].

$$C_e = C_I \left[\left(1 + \left(\frac{r_c - r_i}{n_L} \right) \right)^{X_{nL}} - 1 \right] \tag{1.18}$$

r_c is the interest rate for loaning the money
r_i is the rate of inflation
n_L is the number of times the interest has to be paid a year
X is the design lifetime in years

- **Levelized cost of energy**
A way of representing the costs of wind farm is by using the cost of energy (COE) formulation. Most often used is so call Levelized Cost of Energy (LCoE) expressed

as Equation 1.19.

$$LCoE = \frac{C_I}{20 \cdot AEP} \tag{1.19}$$

- **Net present value**

  As mentioned, the value of money is changing during time. This can be understood as follows: the present value PV, paid in N years with an inflation rate of r is worth FV. The r can simply represent the inflation, but can also include various factors related to risk. Since it reduces the value of money throughout time, it is known also as the discount rate. Using the discount rate, we can compute how much money we have to repay (negative cash flow) in order to return the loan received as the initial investment (year 0). Furthermore, it is possible to compute the behaviour of the value of income (positive cash flow) the farm is generating. Discounted cash flows give the present value, which summed together with an initial investment give the Net Present Value (NPV). This is schematically shown in Figure 1.5. A positive NPV indites the project is profitable with annual return r and additional revenue in the amount of NPV.

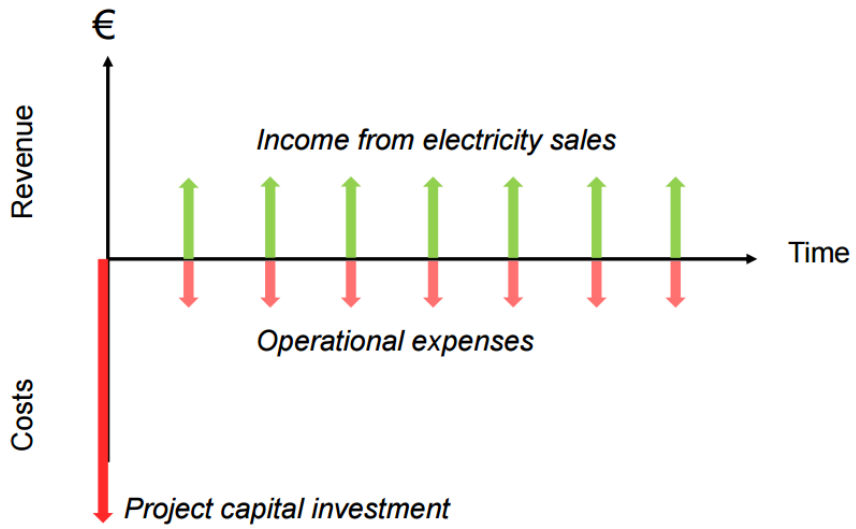


Figure 1.5: *Positive and negative cash flows. DTU Wind Energy, Technical University of Denmark, ECO1:Wind Farm Economics*

Positive cash flow per year is the amount of money from selling the electricity and is given in Equation 1.20. The negative cash flow in Equation 1.21.

$$PF_i = AEP \cdot p_{el} \cdot (1 + r)^{-i} \tag{1.20}$$

where $i = 1..20$ years and the price of electricity p_{el} is approximately 0.37€/kWh (for the time being).

$$NF_i = C_{O\&M} \cdot (1 + r)^{-i} \tag{1.21}$$

where $C_{O\&M}$ is the yearly cost of maintenance.

NPV is given in Equation 1.22.

$$NPV = -C_I + \sum_{1}^{20}(PF_i - NF_i) \tag{1.22}$$

Note there may an interest rate to the loaned money, which will increase the investment in Equation 1.22.

1.4 Required tools for wind farm analysis

From the previous section, it follows the wind farm simulation requires models of power output and load as input for the computation of costs. While stationary flow field models the power production sufficiently well, capturing the fluctuating inflow conditions which dictate the wind turbine loading and derived O&M costs requires the in-stationary modeling of the wind farm flow field [23]. For this reason analysis of individual the wind turbines is performed by HAWC2, which in turn uses the DWM model to simulate the turbines in farm condition.
The computational effort of using HAWC2 as a simulator for purpose of optimisation is too great. Note that each turbine in layout configuration needs to be simulated for 23 wind speeds and 12 wind directions. Furthermore, each new layout requires a separate set of the turbine simulations. Having this in mind and using TOPFARM [22] as an example, it is decided to supply the power/loading as a surrogate model of selected HAWC2 simulation outputs. HAWC2 DWM model simulations outputs considered for this project are the power production and loads at the blade root, shaft and base of the tower. The power output is, along with a wind distribution at the site, used to estimate the annual energy production. Loads are, again with input from wind distribution, used to analyse the fatigue driven degradation of components as a lifetime equivalent load.

From described site parameters it follows the surrogate model is characterised by:

- Mean wind speed Figure 1.6a
- Spacing to nearest turbine Figure 1.6b
- Angle between the ambient wind direction and turbine alignment (from now on termed as "the wake angle") Figure 1.6c
- Wake model
- Turbulence intensity
- Turbine type

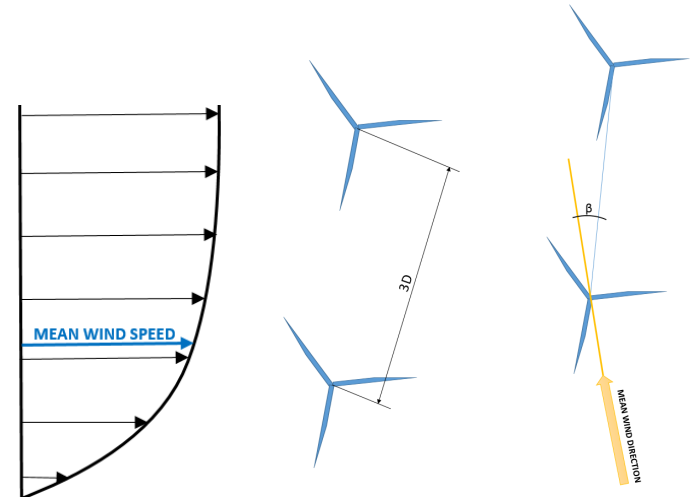
The surrogate model further depends on the shear blocking factor Γ and the turbulence length scale. However, for normal atmospheric conditions and turbines higher than 60m, these remain constant.

Out of these parameters, the wake model is determined by the simulator, i.e. DWM [21] model is used.

The turbulence intensity, and length scale are site parameters. Although it would be very convenient to create a surrogate model with turbulence intensity as an input, required effort might be too great. The surrogate will thus be site-dependant and will use I_{ref} as defined by IEC 61400-1 [1], while the rest of the turbulence parameters will be set as recommended in the HAWC2 manual [25].

The surrogate also depends on the turbine characteristics. In this work, NREL 5MW turbine [15] was used.

In conclusion, the surrogate will be modelled for varying wind speed, spacing and wake angle. Furthermore, such a model will be site dependant, which means it will built based on the turbulence intensity (reference turbulence intensity) at the proposed farm location.



(a) Mean wind speed (b) Spacing to nearest turbine (c) Wake angle

Figure 1.6: *Input variables*

Chapter 2

Surrogate models in application for optimisation of wind farm layouts

2.1 Surrogate modeling explained

- **What is a surrogate model and why do we use it?**
  A surrogate model, as the name suggests, is a substitution of the true response of some computationally expensive or technically demanding system. It is an approximation of the system behaviour on a specified domain range and also in between the sampling points used to create it. The purpose of the surrogate is thus twofold. First, it serves as a cheap, yet accurate, substitution of the true system and second, it is a preferably smooth, continuous, analytic function. This makes surrogate models ideal for fast use in optimisation loops characterised by high number of function evaluations.

- **What types of surrogate models exist?**
  Surrogate models are various regression or interpolation functions. These can be in a form of polynomials [34], space maps [3], neural networks [29] [33] [37], radial basis functions [7] [43], Gaussian processes [35] [28] or support vector machines [41]. A good overview summarising many surrogate models is found in chapter 3 of [18], while the same authors published [17] as a collection of examples, where surrogate models are used on different types of problems. Studies of different models and some advices regarding their application may further be found in [40].

- **How is surrogate model built?**
  The traditional approach of building a surrogate consists of choosing a set of sampling locations where the true function is evaluated. This stage is often referred to as the Design of Experiment (DOE) and is crucial for the successful creation of the model! Once the sampling is done, the type of a surrogate model is chosen and fitted to the data points by optimising the hyper-parameters of the modeling function.
  A more optimal way of creating a surrogate model is by implementing a sequential modeling design. Such an approach involves an iterative procedure of fitting a surrogate model to existing samples and then, based on the behaviour of the fit, choosing new sampling locations. This procedure is repeated until the desired targets are reached. Sequential modeling is also referred to as an active learning process, where the information gathered at the end of each step is exploited to reach some desired output in the next step.
  The sequential modeling is guided by continuously measuring the surrogate during the design phase. It is imperative the measuring functions are defined properly and without bias towards certain trends.
  Building the surrogate model will thus depend on the following:

  1. Choosing the initial set of samples through DOE
  2. Choosing a type of surrogate model
  3. Choosing an algorithm for sequential modeling
  4. Specifying how the surrogate is measured

  These are described in the continuation of this chapter.

2.1.1 Design of Experiment (DOE)

The method of choosing the initial set of sampling locations is made with an intention of evenly covering the domain, while keeping the number of samples relatively low. Each sample is defined by the combination of design variables, each at a certain level. The levels are defined by binning the range the variable covers. Sampling all possible combinations of variables at all levels is termed a full-factorial design. This is the simplest and the most expensive method for DOE. Other known DOE algorithms are Orthogonal Array sampling, Latin Hypercube sampling or Monte Carlo sampling. These are briefly described in the following of the section, but first, the main points regarding DOE are listed:

- DOE is a strategy of allocating samples in the design space [18]

- Purpose of DOE is to reduce the number of experiments while maximising the amount of information acquired through each experiment
- The samples are chosen so to capture the global behaviour of a true function
- DOE is characterised by number of variables and number of levels of each variable

Full-factorial design consists of experiments where all combinations of variables on every level are performed (cf. Figure 2.1). The number of required simulations is thus the product of number of levels for ech variable.

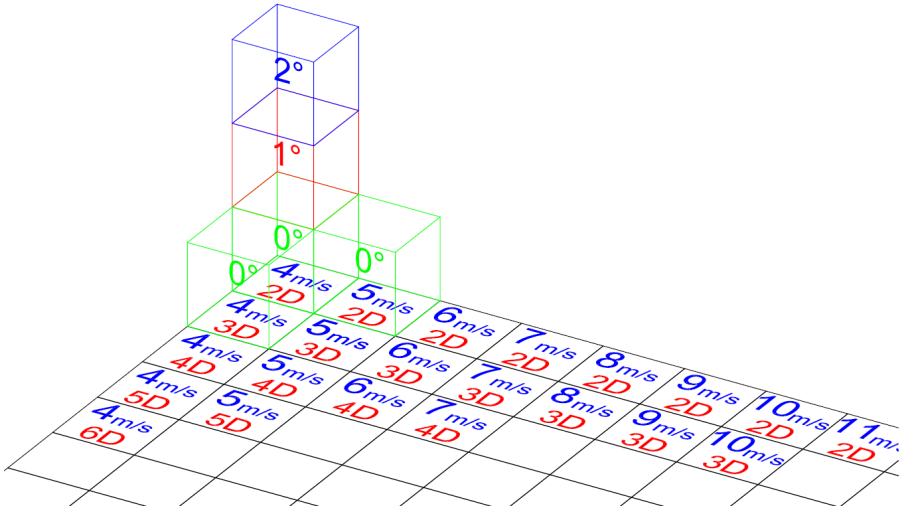


Figure 2.1: *Full factorial design for the surrogate model example. In case of 22 wind speeds ($4m/s$ to $25m/s$), 11 spacings ($2D$ to $12D$) and 51 wake angles ($0°$ to $50°$), full-factorial would require 12342 sampling points (simulations)*

Orthogonal Array (OA) is a matrix of integers describing the coverage of the design space with the purpose of accounting for all variables at all levels. In order to reduce the amount of samples, it encompasses an combinatorial algorithm so to avoid repeating a variable at a level that has already been used. According to [8], this sampling yields uniform coverage in any t-dimensional projection of a k-dimensional design space. (k being number of variables and t defining the strength of OA).
Latin Hypercube sampling (LHS) is a special case of orthogonal arrays where the strength (explained later on) of the array is 1, and as a consequence each variable is represented by the same number of levels (cube). Notation for orthogonal array according to [13] is OA(N, k, s, t) where N is the number of runs, k number of variables, s number of levels and t the strength of the orthogonal array.
To better understand OA and LHS an example is given in Equation 2.1 showing OA(4,3,2,2).

First notice this is a (4 by 3) matrix. The number of variables (columns) is 3. Each variable has 2 levels (0 or 1). The experiment is designed for 4 runs (rows). The strength of this array is 2, since each of possible combinations of levels $((0,0), (0,1), (1,0), (1,1))$ in 2 columns appears only once (1 appearance is termed index $\lambda = 1$).

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} \tag{2.1}$$

Reflecting on this quick intro to OAs, one may ask what happens in case if the variables do not have same number of levels? A reader may further notice OAs do not exist for just any choice of K, n, s and t. In fact, only selected parameters can create a true OA. This is also the reason why no extensive study into subject of OAs and LHS will be given in this work. Readers further interested in the field of combinatorics are warmly pointed to [13] for extensive study in OAs and [32] or [16] for practical construction of OAs or LHS.

Monte Carlo (MC) algorithms are the next popular choice for designing an experiment. There are several different MC approaches for choosing samples such as *pseudo*, *stratified* or *quasi*. Most MC algorithms are based on a random choice of sampling locations. *Pseudo-MC* uses pseudo-random number generation to mimic a truly random process. If the design space is divided in bins of equal probability and then a sample is chosen as a random level in each bin the MC is termed stratified (Figure 2.2). *Quasi-MC*, on the other hand, employs some kind of deterministic algorithm to distribute samples in K-dimensional space.

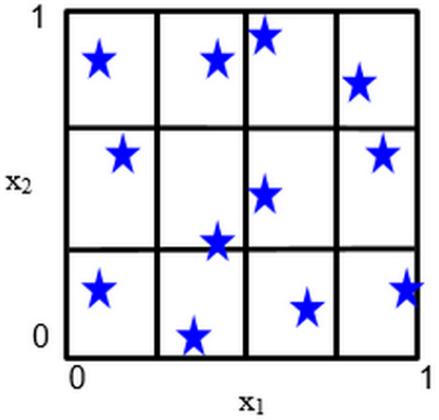


Figure 2.2: *Stratified Monte Carlo sampling where the bins are sized to have equal probability, and a sample is randomly placed in each bin. Reproduced from [8]*

The topic of DOE will not be covered in more detail. When combined with sequential modeling design, determining the initial set of points is not as important as in the case the model would not be subsequently corrected. Nevertheless, number of samples in the initial set as well as domain coverage will, to some extend, have an effect on the modeling process. An example of this will be shown in subsection 2.2.1.

2.1.2 Choice of surrogate model

While most model types manage to fit simple functions reasonably well, systems characterised by variable behaviour over different portions of domain, require the choice of the modeling type to be done more carefully. Desired characteristics of a good surrogate model are:

- Computationally cheap to build
- Reasonably accurate
- Smooth

Surrogate modeling techniques covered in the continuation stand out as examples already found in the literature, but also due to their ease to implement.
Important notion in surrogate modeling are the basis functions. A model equation is a set of basis functions superimposed in various combinations so to closely mimic the behaviour of the true function. Types of surrogate models may differ in the library of basis function they use (e.g. polynomial base, Fourier base). Usually, the main difference is, however, found in additional terms of the modeling equation. Models may further be distinguished as interpolation schemes, where the response of the surrogate at the sampling location has equal value as the true function, or as regression schemes, where this may not be the case.

`Polynomial regression`

The most basic modeling technique is polynomial regression, which approximates the system by a sum of weighted functions as seen in Equation 2.2, i.e. the modeling function is a n-th degree polynomial.

$$f(x) = \sum_{j=1}^{N} \beta_j g_j(x_i) \tag{2.2}$$

where
g are the modeling functions (e.g. linear/quadratic/cubic fit) evaluated at sampling locations
x are the sample locations
β are the weighting coefficients

The fitting is done by optimising the weighting coefficients in order to minimise, for example, the sum of residuals squared (LSQ or l_2 estimation), or the maximal residual (l_∞). From the point of view of optimisation, this may be a defined as a linear or quadratic problem, for both of which an analytic solution exists. Example for LSQ estimation and its formulation as quadratic program is given in Equation 2.3.

$$\begin{aligned} min_x \quad & f(x) \\ & f(x) = \tfrac{1}{2}\left\|y-\hat{y}\right\|_2^2 = \tfrac{1}{2}\left\|y-Ax\right\|_2^2 \\ & f(x) = \tfrac{1}{2}(y-Ax)^T(y-Ax) \\ & f(x) = \tfrac{1}{2}(y^Ty - y^TAx - x^TA^Ty + x^TA^TAx) \\ \texttt{since} & \\ & x^TA^Ty = (Ax)^Ty = (y^T(Ax)^T)^T = (scalar)^T \\ & f(x) = \tfrac{1}{2}x^TA^TAx - y^TAx + \tfrac{1}{2}y^Ty \\ & f(x) = \tfrac{1}{2}x^THx + g^Tx + \rho \end{aligned} \tag{2.3}$$

where
$\|x\|_2^2 = \sqrt{\sum_{i=1}^n x_i^2}$ is the 2-norm squared $H = A^TA$
$g = -A^Ty$
$\rho = \frac{1}{2}y^Ty$

$$A = \begin{bmatrix} t_i & 1 \\ \vdots & \vdots \\ t_m & 1 \end{bmatrix} \tag{2.4}$$

Notice the ρ term in Equation 2.3 does not affect the minimisation of this equation, since it is a constant. Thus the input for minimisation are only H and g, where A is defined as Equation 2.4. Equation 2.3 is a quadratic equation, which can be solved using Quadratic Programing (QP). Formulated as normal equation (A has to have full rank, i.e. number of

basis functions is less than number of data points), the solution is given as in Equation 2.5.

$$\begin{aligned} &\nabla f(x) = 0 \\ &(A^T A)x = -g^T x \\ &Hx = -g^T \\ &x_{opt} = -H^{-1} g^T \end{aligned} \tag{2.5}$$

Radial Basis Functions

Radial basis functions interpolation scheme uses a linear sum of radially symmetric functions such as for example $g(r) = r$, $g(r) = r^3$, $g(r) = r^2 \log(r)$. A function is radial if it satisfies $g(x) = g(||x||)$, where the distance is usually Euclidean. The associated model equation is given in Equation 2.6.

$$f(x) = \sum_{j=1}^{N} \beta_j g_j(||x - c_j||) \tag{2.6}$$

where $||x - c_j||$ is the argument of function g and c_j are the known function centres.

The model parameters β_j are determined in the same way as for polynomial regression, i.e. by for example fitting in LSQ sense.

Kriging

Kriging is a modeling technique based on Gaussian processes [18]. Kriging models, in addition to the basis functions, encompass noise terms which is justified as a way of mimicking the sample path of a stochastic process [28]. Kriging model equation is given in Equation 2.7.

$$f(x) = g(x)^T \beta + Z(x) \tag{2.7}$$

where 1st term on r.h.s. is equivalent to the polynomial regression, and $Z(x)$ represents the added noise with zero mean and variance σ^2.

The regression has the goal of capturing the global function behaviour, and the noise takes into account the local variations [18]. Kriging is modeled by determining the correlation parameters θ_k in Equation 2.8, where $R(x, y)$ is the Gaussian correlation function used to estimate the covariance of the noise given in Equation 2.9. Model fitting is performed with the aim of finding maximum likelihood for θ [18].

$$R(x, y) = exp[-\sum_{k=1}^{n} \theta_k |x_k - y_k|^2] \tag{2.8}$$

$$Cov[Z(x_i)Z(x_j)] = \sigma^2 \mathbf{R}([R(x_i, x_j)]) \tag{2.9}$$

More details regarding Kriging predictor may be found in [18], [28].

Artificial Neural Networks

Recently, systems for artificial intelligence are becoming increasingly popular. Although such ideas have existed for some time now, greater computational possibilities, but also the complexity of problems engineers are dealing with these days, have turned more attention to machine learning concepts. The author is not yet at this stage very familiar with these systems enough to explain them in detail, however, examples of their use found in various publications certainly make them a very interesting topic to study. Dr. Volker Tresp from Ludwig Maximillian University in Munich describes that the machine learning process can be divided in three groups [33]:

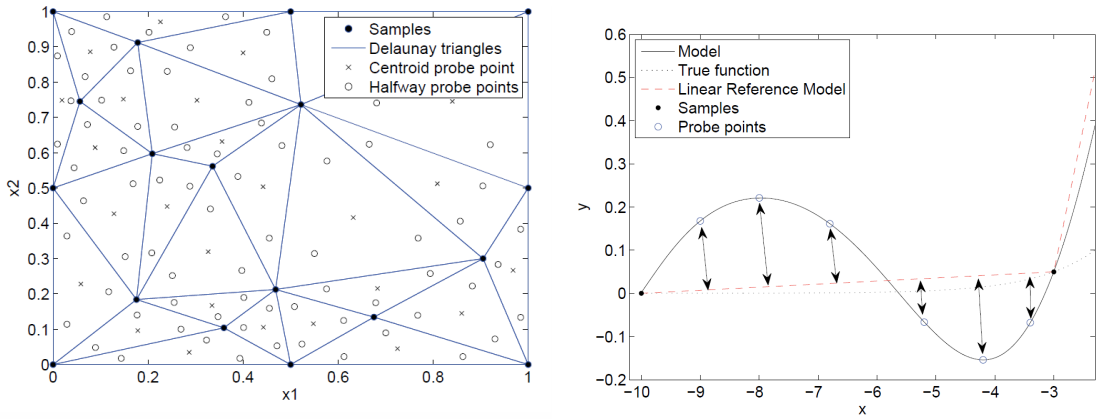
1. memorisation (such as the ability to remember facts)
2. skills (such as the ability to learn to throw a ball)
3. abstraction (such as the ability to form rules based on observations)

The sequential design is a smart algorithms, however, it follows a priori user-defined rules. A system representing an artificial intelligence (AI) where the algorithm has the ability to create its own rules are the Artificial Neural Networks (ANN).
Reason why ANN are mentioned is their already existing application in predicting demand/supply of energy [37]. Their use is for sure to be even more extended with more renewable energy sources being introduced into the grid.
Artificial neural networks work by establishing connections between input data and observed results. This is done using weighting coefficients, basis functions, but also complex "cause and consequence" diagrams. Often, this results in considerable amount of `if-then`, and **case** statements. The main concept in ANN is learning. In the beginning, the model is simple, and its prediction of the output result is often poor, but as the system grows, more connections between cause and consequences are established, and the system is able to give very accurate predictions. A unique advantage is that ANN are able to make a reliable prediction by quantifying even the unknown parameters.
However, ANN are very slow on account of their complexity. They accept very large number of input variables. This allows them to account for some unknown influences and model even the most non-linear functions. The ANN are ideal as a live monitoring tool that learns over time, and when sufficiently trained, can be used for prediction and

optimisation.
Finally, it is reasoned that for the purpose of this project, the use of ANN as the surrogate model type is not necessary, and a simpler model should be used.

2.1.3 Sequential Modeling and Active Learning

In the initial stage, sampling is done with the aim of covering the domain and capturing the global trends of the system. The size of samples in the initial step is kept small, thus, the fitted surrogate is termed coarse. Based on the behaviour of this surrogate, domain coverage and the error size to the true response new set of sampling locations is determined. This procedure is repeated in sequential steps as long as the surrogate does not begin to show desired characteristics. Choosing the new sampling location is done as a compromise of two strategies: Exploration and Exploitation.
Exploring the domain means the new points will be sampled in the parts of the design space which is scarcely populated with data points (Figure 2.3a).
Exploitation of the surrogate is done by choosing sampling locations based on (non)linearity of the surrogate model (Figure 2.3b) [6] [31], but may also be done w.r.t. to the error towards the true function.



(a) 2D Delaunay tesselation of the input space. The location of the probe points in the input space are denoted by the circles and crosses. The location of the samples are denoted by the black dots.

(b) Intuitive 1D illustration of the LRM (Linear Reference Model) measure. The goal is to minimize the deviation between the surrogate model and the linear fit. The LRM measure penalizes the models proportionally to the distance between a number of probe points and the linear fit (these distances are denoted by the arrows).

Figure 2.3: *Exploration and exploitation. Reproduced from [31]*

2.1.4 Measuring the surrogate model

Defining the measuring methods properly is important for the purpose of evaluating the quality of the surrogate model, but also to guide the convergence towards an desiered optimal design. Some popular measures of error are summarised in Table 2.1.

Table 2.1: *Error measures. y is the measurement, $\tilde{y}$ is the estimate, $\bar{y}$ is the mean*

Error	Type	Formula
Mean Square Error (MSE)	absolute	$\frac{1}{n}\sum_{i=1}^{n}(y_i-\tilde{y}_i)^2$
	relative	$\frac{1}{n}\sum_{i=1}^{n}(\frac{y_i-\tilde{y}_i}{y_i})^2$
Root Mean Square (RMS)	absolute	$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i-\tilde{y}_i)^2}$
	relative	$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\frac{y_i-\tilde{y}_i}{y_i})^2}$
Average Euclidian (AE)	absolute	$\frac{1}{n}\sum_{i=1}^{n}\sqrt{(y_i-\tilde{y}_i)^2}$
Geometric Average (GE)	absolute	$(\prod_{i=1}^{n}\sqrt{(y_i-\tilde{y}_i)^2})^{\frac{1}{n}}$
Harmonic Average (HA)	absolute	$\left(\frac{1}{n}\sum_{i=1}^{n}\frac{1}{\sqrt{(y_i-\tilde{y}_i)^2}}\right)^{-1}$
Maximum Relative (MR)	relative	$\max\left(\frac{y_i-\tilde{y}_i}{y_i}\right)$
Maximum Absolute (MA)	absolute	$\max\lvert y_i-\tilde{y}_i\rvert$
Root Relative Square Error (RRSE)	relative	$\sqrt{\frac{\sum_{i=1}^{n}(y_i-\tilde{y}_i)^2}{\sum_{i=1}^{n}(y_i-\bar{y}_i)^2}}$
Bayesian Estimation Error Quotient (BEEQ)	relative	$\left(\prod_{i=1}^{n}\frac{\sum_{i=1}^{n}\lvert y_i-\tilde{y}_i\rvert}{\sum_{i=1}^{n}\lvert y_i-\bar{y}_i\rvert}\right)^{\frac{1}{n}}$

Common measures of error are mean square error (MSE), absolute average error (AAE) and the most widely used the root mean square error (RMS). However, they depend on units, and due to their absolute nature they tend to ignore small while penalising large values. These measures are thus called pessimistic. The advantage to some other measures is that these functions are quite easy to interpret. Another absolute error measures is the average Euclidean error (AEE), which is the Euclidean distance to the true value (often misinterpreted for RMS). Downside of AEE is that it suffers from depending too much on the outliers.

Examples of "optimistic" measures (dominated by small values) are, for example, geometric (GA) and harmonic average (HA) errors. From the equation in Table 2.1 it can be seen the main disadvantage of GA error is that it fails if any of the estimates matches the measurement (0 value at one point causes the whole measure to become 0). HA error is,

according to [11], suited best for cases where the the error fluctuates over different runs (application with cross validation is mentioned).
In general, absolute errors suffer from the fact the tolerance (for the stopping criteria) is hard to set up beforehand, and from the fact they are dependant of the units.
Relative errors are usually preferred, since they measure global error behaviour (w.r.t. some reference value). Since relative error formulation come in a form of a fraction, special attention must be given to denominator. In case the denominator is very small, the value of error might be misleading. Problems such as this are solved by adding a constant to the denominator or translating the response. However, such absolute/relative hybrids might be hard to interpret.
A good measure of relative error is the Root Relative Square Error. It measures the deviation w.r.t. to the simplest fitting model, the mean. Like with most of the error functions, one needs to understand the behaviour of the response to use RRSE properly. In case where the mean is a good fit to the values, the value of the error might give pessimistic estimate. Furthermore, interpretation of RRSE as well as determining the stopping criteria might be difficult.
The Bayesian Estimation Error Quotient (BEEQ) is another relative error good for being less sensitive to large values, however, suffers the same problem as the GA.
In general, relative error measures are preferred.
The error functions mentioned above may only be used on regression models, since their application for interpolation scheme makes no sense. For this reason it is common to split the samples into two sets. The training set is then used to fit the model, an the validation set is used for measuring purpose.
Apart from the validation set, common approach is to use the cross-validation. This method quantifies the error by excluding samples from the modeling process similar to the validation set. The difference is that in cross-validation, all samples are excluded through various combinations. The measure of k-fold cross-validation is computed as a average of errors resulted from all possible sets of k samples excluded. (Excluding k samples from the training set defines a k-fold cross-validation.) For a set of n points, k-fold cross-validation requires $\frac{n!}{(n-k)!}$ evaluations. It can be seen the cross-validation analysis is quite expensive.
In conclusion, defining the measures for the surrogate model has to be done by carefully examining pitfalls and ensuring no bias occurs. To achieve this, it is necessary to implement a multiobjective surrogate evaluation encompassing several measures at the same time. This way downsides of each individual measuring technique is reduced.

2.2 SUMO - Introduction to the toolbox

For the purpose of this work SUMO toolbox [10] is used. This is a free (for academia) toolbox developed by SUMO (SUrrogate MOdeling) Lab, which is part of the IBCN research group of the Department of Information Technology (INTEC) at Ghent University - iMINDS.

SUMO toolbox is a combination of many functions and segments of code written in MATLAB, C, and Java. The main interface for set-up is through a **.xml** file. The toolbox is built in a modular way thus enabling any additions, such as algorithms or functions, to the existing framework. This however, requires knowledge of object-oriented programming in MATLAB.

Purpose of the following section is to introduce the reader to the SUMO toolbox and also to show the author's learning process. The section will mention important parameters required to properly set-up the toolbox and give simple examples of some modeling options. For this purpose tests are performed on a 2D Rosenbrock function (Equation 2.10, Figure 2.4). Although Rosenbrock is usually used for testing optimisation algorithms, it will suffice for the intended purpose.

$$\underset{x\in\mathbb{R}^2}{} \quad f(x) = (100x_1^2 - x_2)^2 + (x_1 - 1)^2) \tag{2.10}$$

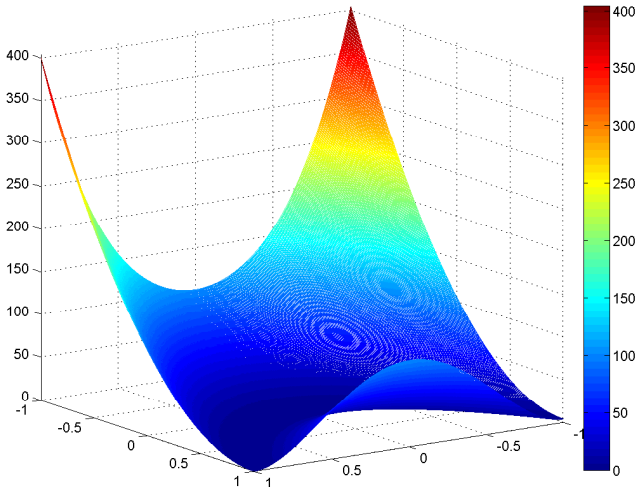
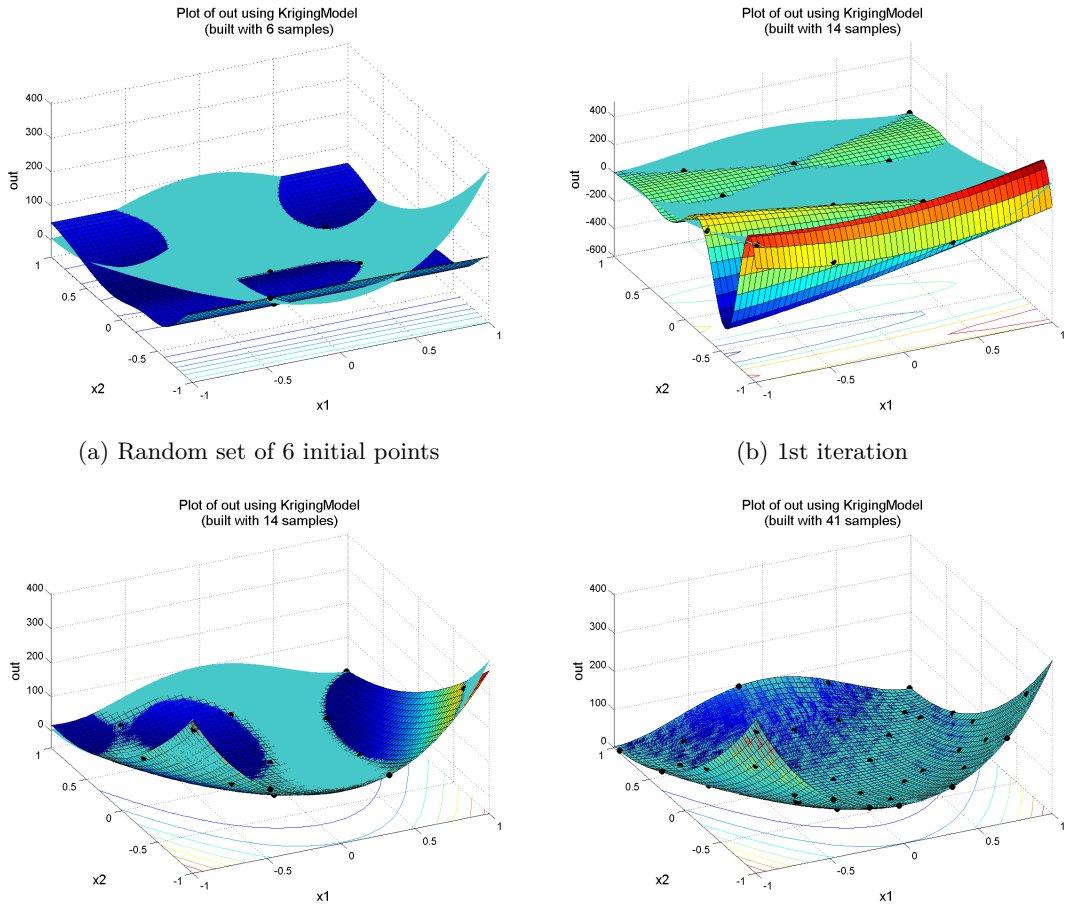


Figure 2.4: *2D Rosenbrock function*

2.2.1 Initial Design

As mentioned, modeling of the surrogate is done using a sequential approach, i.e. sampling is done in a number of steps based on specified criteria. As a consequence, the initial set of points successively looses on significance. Nevertheless, this section will illustrate the effects of making different initial sampling. Parameters of importance for the DOE phase are the position and the number of samples.
Reader should note the rest of the modeling settings are set to optimal values and have no influence to the results presented next. For the first test, the initial set consisting of 6 randomly chosen points is used. Steps until the final surrogate model is reached are shown in Figure 2.5.
Notice the 1st and 2nd iteration (1st iteration here means one run after the initial samples) use same points, but the model is clearly different. This is due to the difference in the use of basis functions to fit the data. Apart from the hyper-parameter optimisation, modeling of the surrogate is done by varying the basis functions until the optimal modeling equation is found. This may be done either by iteratively creating new surrogate from scratch, or by superimposing basis functions to the previous model. SUMO also offers the use of an evolutionary algorithm for the purpose of finding the optimal modeling eqution.
Going back to Figure 2.5b, the surrogate model shows wrinkles which indicate a bad fit (smoothness is preferred!). A better model, using different set of base functions, is obtained in the 2nd iteration as shown in Figure 2.5c. This example gave an insight how the SUMO chooses the modeling equation and indicated a possible issue in case the global trends are not captured in the start of the modeling.
However, the point of this section is to show the behaviour w.r.t. position and number of initial samples. Compared to Figure 2.5 where only 6 initial points were used and total of 41 samples were required to reach the set tolerance, initial set of 16 samples (Figure 2.6) reduced the total number of required samples to 34. Even before other test results are commented on, the importance of setting sufficient number of initial sampling points for any method of DOE is clear.



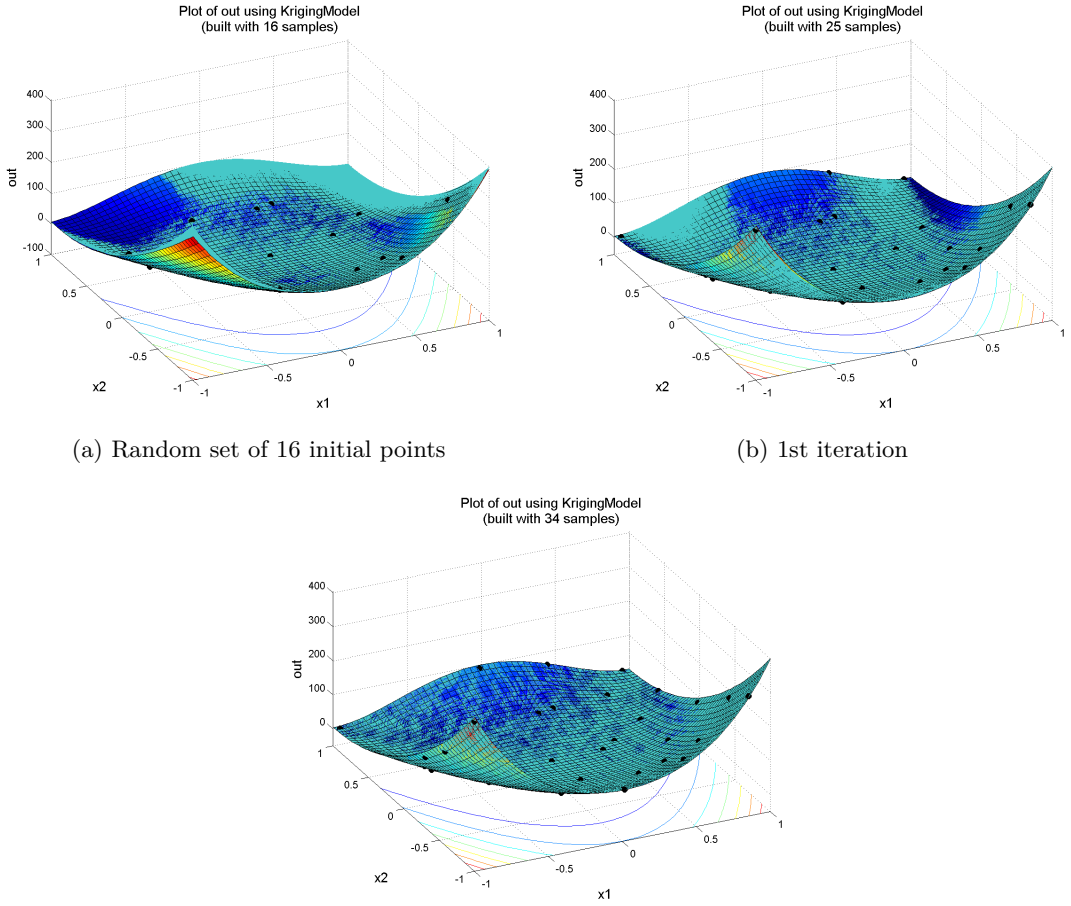
(a) Random set of 6 initial points

(b) 1st iteration

(c) 2th iteration

(d) 5th iteration (final)

Figure 2.5: *Sequential surrogate modeling of 2D Rosenbrock function. Initial set are 6 randomly chosen points. True model is shown in pastel blue colour.*



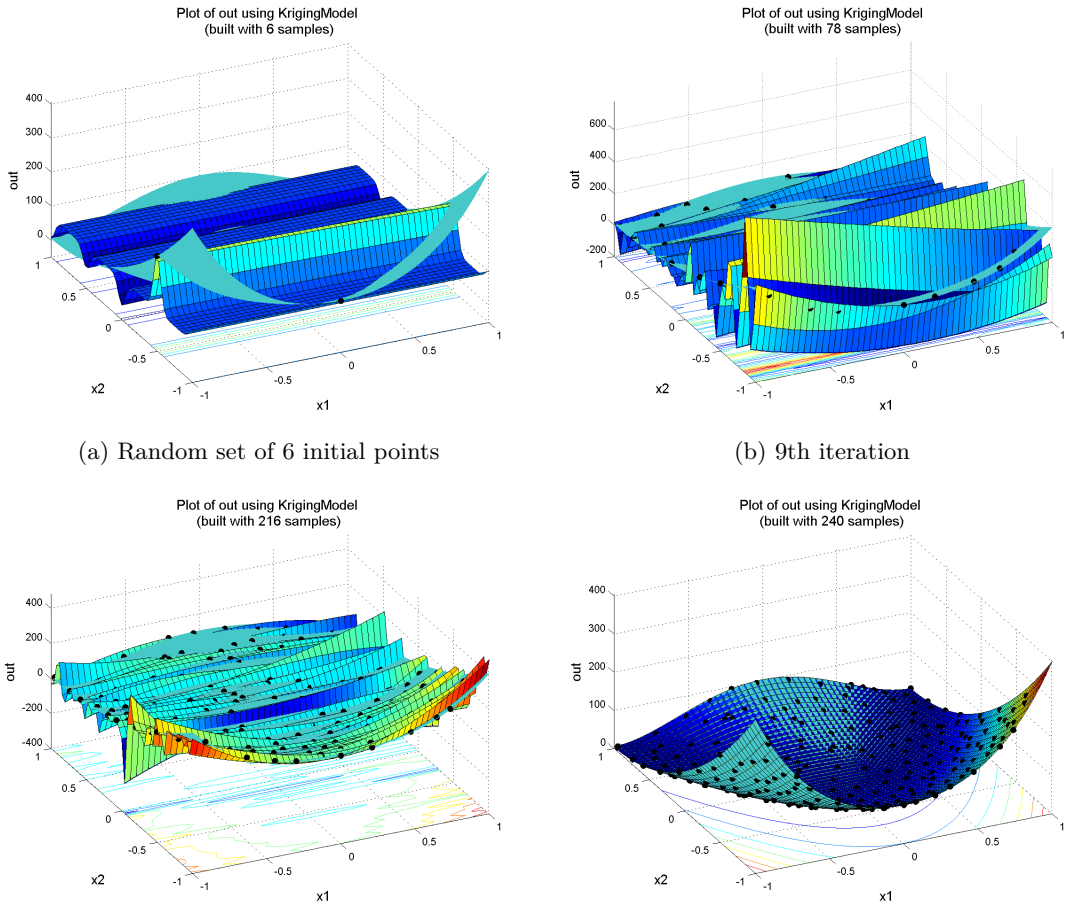
(a) Random set of 16 initial points

(b) 1st iteration

(c) 2th iteration

Figure 2.6: *Sequential surrogate modeling of 2D Rosenbrock function. Initial set are 16 randomly chosen points. True model is shown in pastel blue colour.*

Next example in Figure 2.7 shows the extreme case of LHS where the global behaviour of the true function was wrongly predicted. Due to this, the approximating model diverges from the true function, and good fit is obtained only after a high number of points has been evaluated. Furthermore, if surrogate model is build on top of the previous model and not started from scratch after new samples are supplied, model will result in superposition of too many base functions. This is known as over-fitting.



(a) Random set of 6 initial points

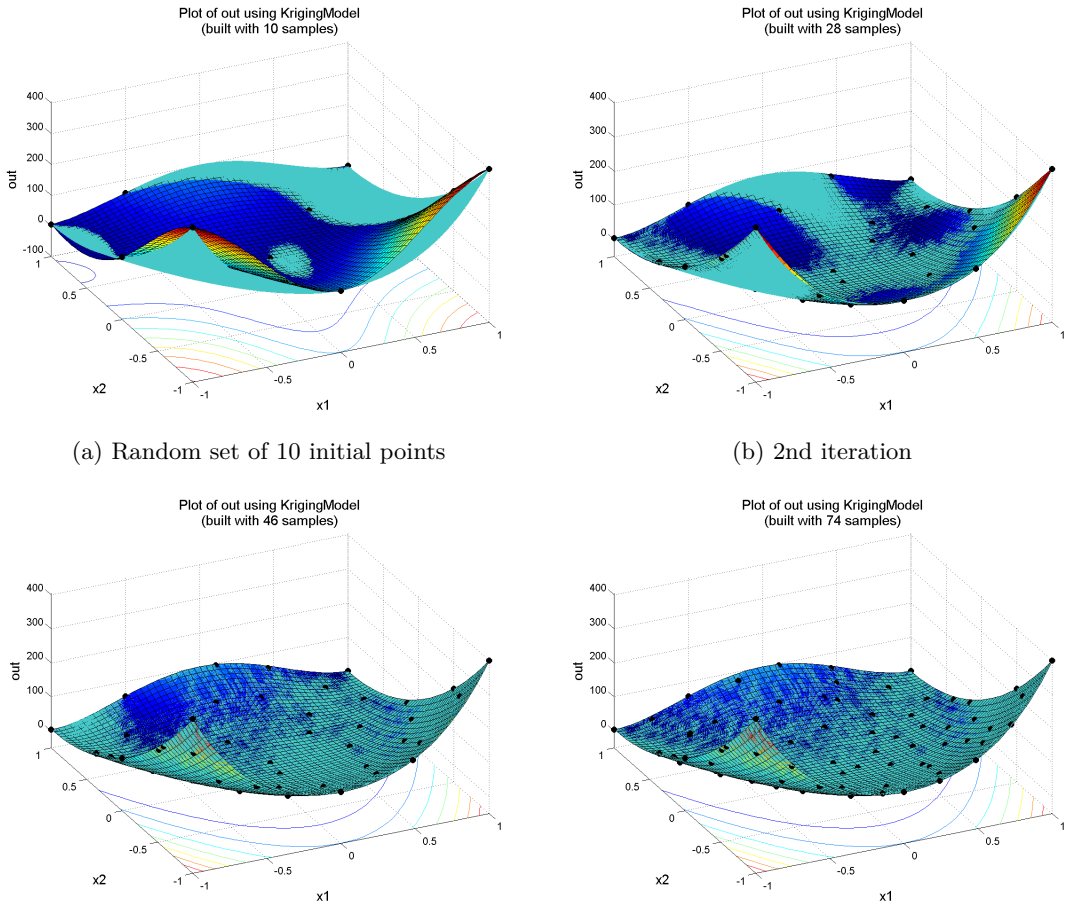
(b) 9th iteration

(c) 24th iteration

(d) 28th iteration (final)

Figure 2.7: *Sequential surrogate modeling of 2D Rosenbrock function. Initial set are 6 points determined by LHS. True model is shown in pastel blue colour.*

Results of the same test, however with the addition of data points sampled in the corner of the domain, are shown in Figure 2.8. No divergence from the true function is observed, thus, the importance regarding the sampling locations and equal domain coverage is confirmed.



(a) Random set of 10 initial points

(b) 2nd iteration

(c) 4th iteration

(d) 7th iteration (final)

Figure 2.8: *Sequential surrogate modeling of 2D Rosenbrock function. Initial set are* $6 + 4$ *points determined by LHS combined with corner points. True model is shown in pastel blue colour.*

To asses which method of DOE is optimal for this problem, a set of 50 initial points was used without allowing sequential design (one-shot design). Both random, LHS or any other kind of algorithm show good results for fitting 2D Rosenbrock problem with 50 sample points. If a sequential design is allowed (still keeping initial set of 50 samples) a randomly chosen set shows same results as LHS, and other methods are very close. Note, however, that choice of sequential design is an influencing factor here.

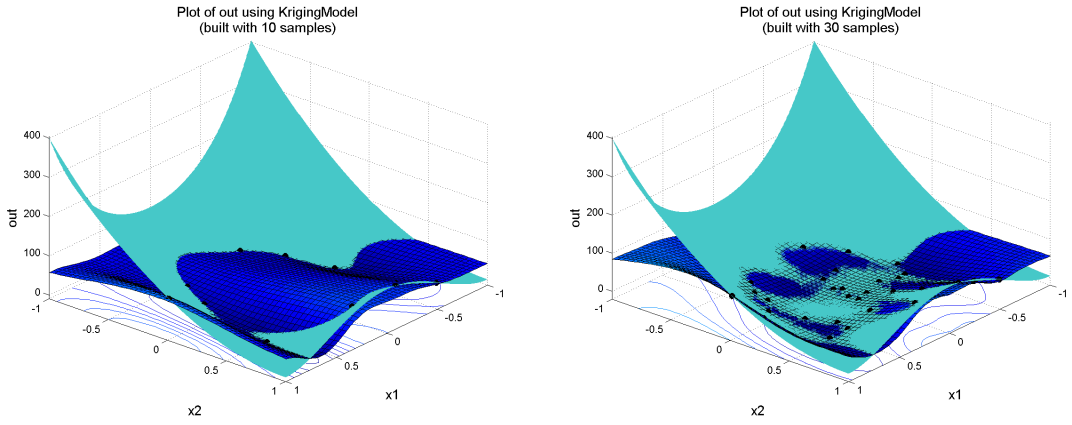
The conclusion is thus that the quality of the surrogate model will depend less on the type of algorithm for choosing initial set of points, if the initial set of samples is sufficiently large and evenly spread over the domain. For a black-box simulator whose true function is expensive to evaluate, the author reasons it is better to specify higher number of initial samples than risk the chance of wrongly estimating the model and causing the solution to diverge. In addition, well specified, multi-objective, measuring of the model should remove

the possibility of making wrong predictions. The main goal of the initial set is thus to capture the global behaviour of the true function.

2.2.2 Algorithms for sequential design

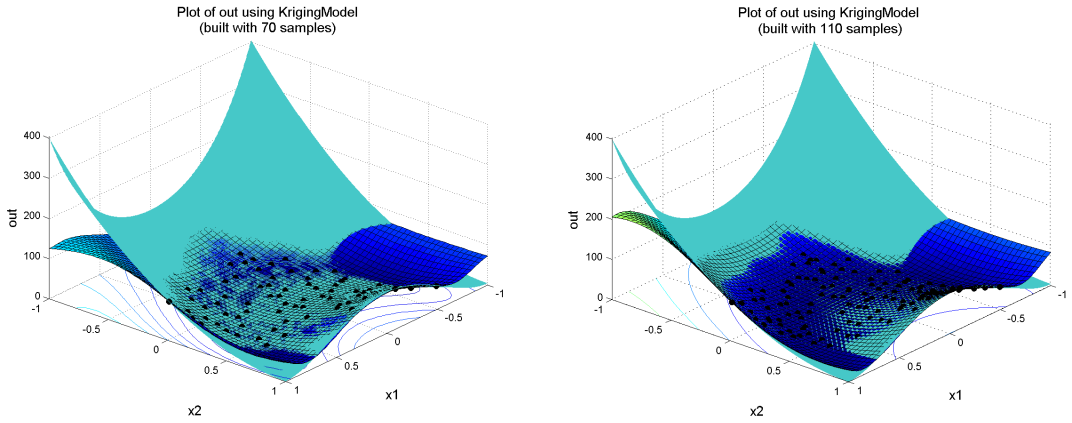
A way of reducing the computational effort required to build a surrogate is by implementing a smart algorithm for choosing sampling locations. Tests which follow will analyse the algorithms for sequential design.

The initial design will be a set of 10 randomly chosen points. The intention is to start with an sub-optimal set in order to examine how sequential modeling performs. As mentioned in subsection 2.1.3 sequential design should be a compromise between domain exploration and surrogate exploitation. First test is made using Delaunay algorithm (see Figure 2.9). Delaunay algorithm uses QHull algorithm to compute the convex hull which is then discretized by Delaunay triangulation. The samples are the selected in locations far from previous sampling points, or in locations where the estimated model error is largest [10]. Naturally, the choice of which error to use for this estimation is the a important parameter.



(a) Initial set of 10 randomly chosen samples

(b) 3rd step



(c) 7th step

(d) 11th step

Figure 2.9: *Sequential surrogate modeling of 2D Rosenbrock function. Delaunay sequential design. Initial set of 10 randomly chosen points. True model is shown in pastel blue colour.*

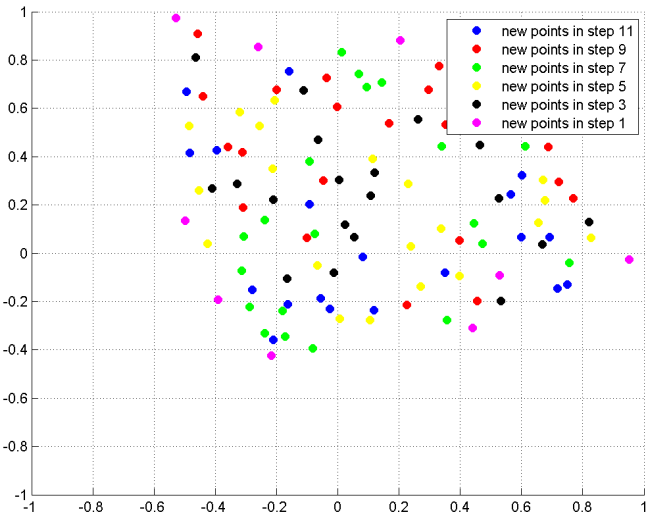
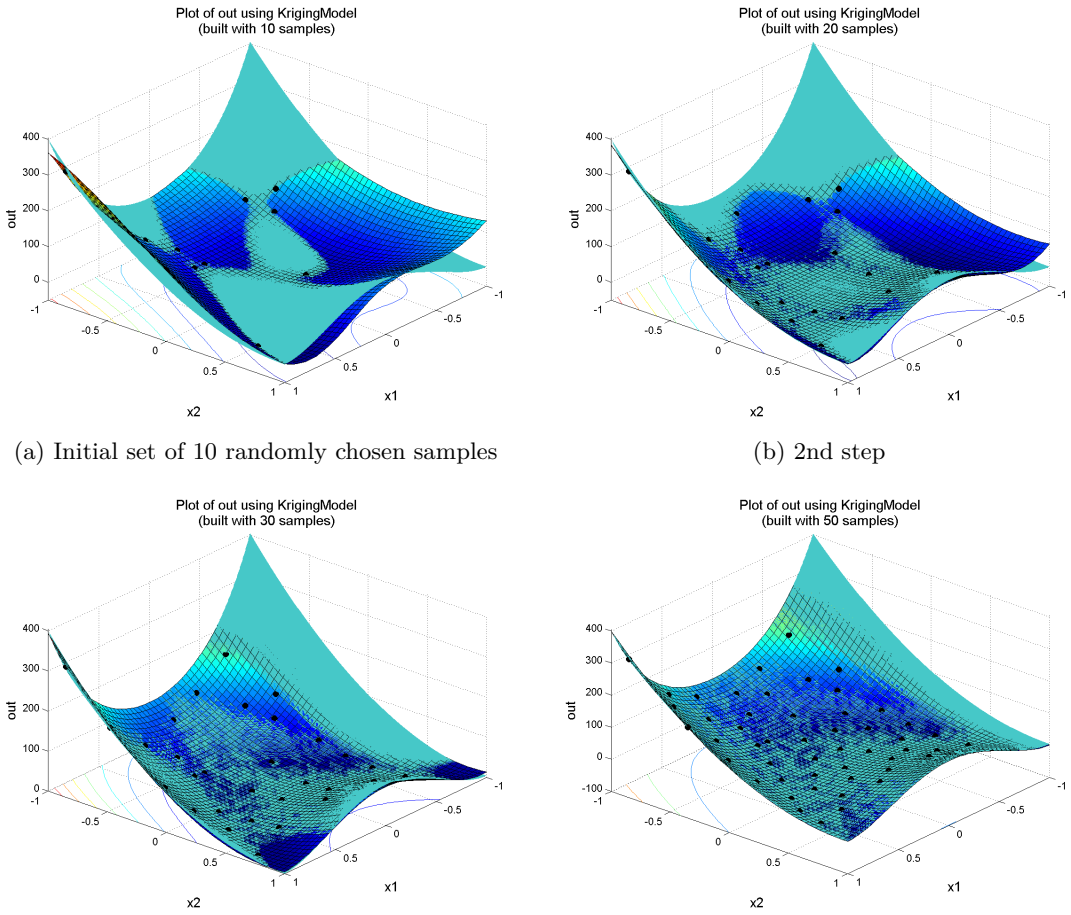


Figure 2.10: *Sampling points for Delaunay sequential design algorithm*

Notice from Figure 2.10 the Delaunay algorithm fails to take any samples near domain edges which causes a poor fit of the model at the bounds. Possible source of this error might lie in the poor initial design, which does not cover the whole domain. Changing to factorial sampling, which covers the domain better, improvement is made in terms of global fit (near edges). Nevertheless, the total required number of samples is even higher (179 sampling points). Thus, the conclusion is that Delaunay performs poorly for this problem. One of the reasons for this is found in the fact the Delaunay makes no analysis of the non-linear function behaviour, only error and domain coverage. Clearly, this is not enough. The reason why the domain was not covered over entire range remains unknown due to the black-box nature of the code.

Next test was made using **density** sequential design. As the name suggest, this is a space filling algorithm based on Voronoi tessellation [10]. Voronoi tessellation is a way of partitioning or discretizing design space based on the distances to certain points. This algorithm outperforms Delaunay by far, which comes as a surprise knowing no error analysis is done prior to making the choice of sample locations. Results are shown in Figure 2.11 and Figure 2.12.



(a) Initial set of 10 randomly chosen samples

(b) 2nd step

(c) 3rd step

(d) 5th step (final)

Figure 2.11: *Sequential surrogate modeling of 2D Rosenbrock function. Density sequential design. Initial set of 10 randomly chosen points. True model is shown in pastel blue colour.*

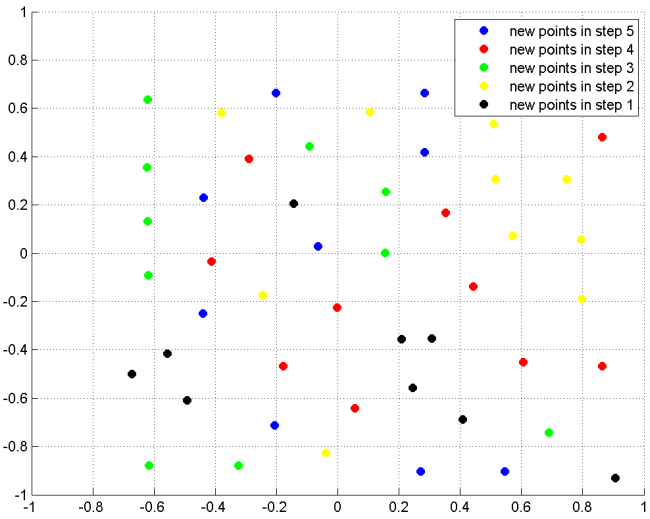
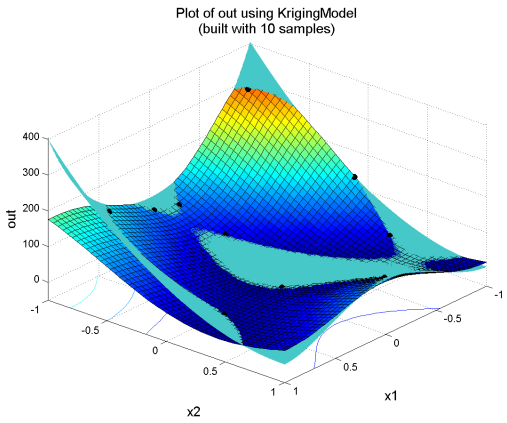


Figure 2.12: *Sampling points for* ***density*** *sequential design algorithm*

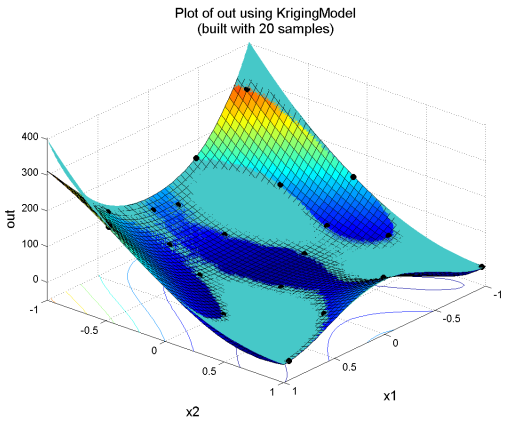
Following example shows the **LOLA-Voronoi** algorithm [6] (Figure 2.13). LOLA (LOcal Linear Approximations) bases the sampling choice around non-linear parts of the surrogate model.

Settings for **LOLA-Voronoi** algorithm include the number of neighbouring points for estimation of non-linearity (LOLA segment) and the choice of method for computing the distance between points (Voronoi segment). The estimation of non-linearity is done by analysing the gradient of the function at a point. Since the gradient is not available analytically (true function is unknown) and the samples are not distributed evenly, application of finite difference for derivative estimation is not possible. The non-linearity is analysed by performing a linear regression fit (LSQ) using neighbouring points and analysing the error to samples [5]. Where this difference is large more points are sampled. Note that if the points are spaced too far away, result might be misleading, since some local fluctuations might be overlooked. In certain cases only couple of neighbouring points might result in better identification of non-linear behaviour. Nevertheless, there is no universally best solution, and again the performance will depend mostly on the initial set of samples.

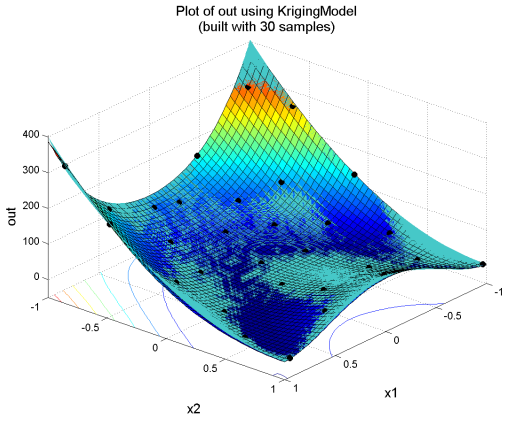
Distances between the points are computed as Euclidian.



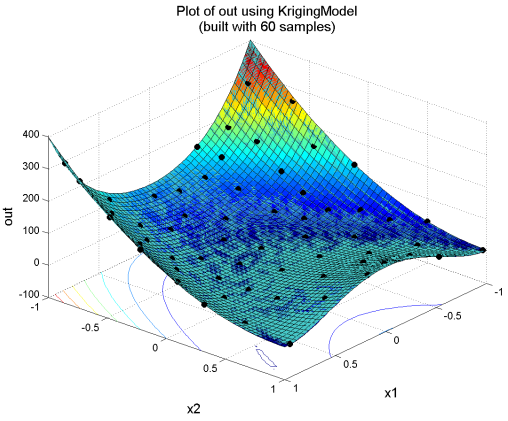
(a) Initial set of 10 randomly chosen samples



(b) 2nd step



(c) 3rd step



(d) 6th step (final)

Figure 2.13: *Sequential surrogate modeling of 2D Rosenbrock function.* ***LOLA-Voronoi*** *sequential design. Initial set of 10 randomly chosen points. True model is shown in pastel blue colour.*

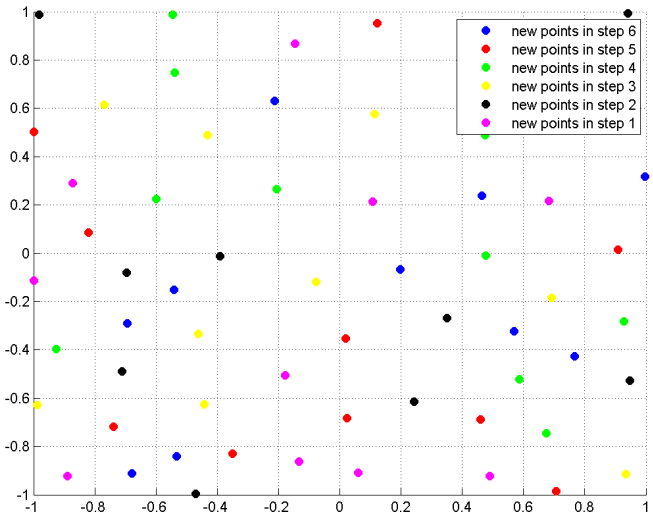


Figure 2.14: *Sampling points for* ***LOLA-Voronoi*** *sequential design algorithm*

Notice from Figure 2.14 that the domain is, compared to Figure 2.12 and Figure 2.10, far better covered after 6 steps. Notice also from Figure 2.13 the surrogate model fits the analytic function much better in the corners of the domain. This was expected knowing both non-linearity and domain coverage are taken into account (exploitation and exploration). SUMO has an option of creating a sequential design algorithm as **combineSequentialDesign** type. This means that several algorithms are used to determine the new set of samples. The algorithm presented as **default** option in SUMO toolbox represents a combination between LOLA-Voronoi and **error** based sampling. In the tests performed, this algorithm has optimal performance.

2.2.3 Available surrogate models

Following models are available as a part or addition to SUMO toolbox.

Regression models

- Rational function models
- Radial basis functions

Gaussian Process models

- DACE (Design and Analysis of Computer Experiments) model - MATLAB Kriging toolbox (developed at DTU)
- Kriging

Spline function models

Artificial Neural Networks

Support vector machines

Ensemble and heterogenetic models

2.3 Building a Dynamic Wake Meandering surrogate model

Following section tries to explain the surrogate modeling settings chosen for the creation of the DWM surrogate model. While the discussion in the previous section was kept general regarding the topic of surrogate modeling, this section focuses on the surrogate targeted for the specific application in WFLO. In addition to the set-up of the modeling process, a surrogate model for a low intensity turbulence site is built and analysed.

The surrogate model models the following outputs:

- Equivalent load (Mx) at the blade root
- Equivalent load (My) at the blade root
- Equivalent load (Mx) at the tower bottom
- Equivalent load (My) at the tower bottom
- Equivalent load (Mx) at the shaft
- Equivalent load (Mz) at the shaft
- Power production

The input variables for each of these are as explained in section 1.4. Furthermore, note the following:

- To allow general conclusions and non-dimensional comparison the values of both power production and loads are normalised by the results of a turbine operating in free wind conditions at 10m/s.
- Each output represents one surrogate model and may be modelled using different model type. Nevertheless, with the aim of keeping the computational costs to minimum, all models are build using same set of samples. Because the outputs most likely exhibit different behaviour, this solution is sub-optimal. However, the smart algorithm used to perform the sampling makes sure each output contributes equally to the choice of new points.

Although many settings may be reasoned based on the theory provided in previous sections, due to high computational costs, insufficient knowledge about the behaviour of the outputs, but also in order to make sure the whole set-up is done properly, modeling has been performed in 3 stages:

1st stage - Sampling at 500 random locations and creation of the coarse surrogate

2nd stage - Sequential surrogate modeling

3rd stage - One-shot tuning resulting in high fidelity surrogate

Before proceeding to details explaining each phase, next section will present the experimental set-up for surrogate creation.

2.3.1 Experimental Set-up

No practical (laboratory) experiments are part of the presented project. Nevertheless, building a surrogate model requires numerous computations, simulations and data processing. From the start of the project, the aim was to create a framework for optimisation of wind farm layouts, which requires as few inputs from the user as possible. Furthermore, the platform was planned in a modular fashion, so that parts of it may be substituted by revised modules without the need to change the general layout. Keeping these targets in mind, a careful planing of the simulation set-up was required.
As a way of speeding up the process of creating the surrogate, it is necessary to make sure large number of simulations may be ran at the same time (in parallel). For this purpose the DTU Gorm cluster was used. Gorm has around 900 CPUs (900 simulations in parellel). Each sample is estimated to take around $30min$ of computational run-time (equivalent to the the length of simulation). Additional time is expected for distribution of the necessary files to the cluster, copying of the result files back to the main directory and post-processing of the results. Each wind speed will also require creation of turbulence boxes (9 files for each wind speed). These will be created in the beginning and copied to cluster nodes when required.
Not requiring user supervision, robust modular design, and the use of the cluster for parallel computing as a consequence requires an interaction between numerous tools (Table 2.2), each in a different programming language. To ensure an uninterrupted execution and continuous flow of information between the sub-modules, a well established coordination is imperative. This task is solved using MATLAB. This means the preparation of all input files, pre-processing, post-processing, running of external commands and programs, and some computations are handled by scripts and functions written in MATLAB.

Table 2.2: Tools

Tool	Task description
MATLAB	Main software for control and coordination
HAWC2	Aeroelastic simulation of wind turbine in wake conditions
SUMO	Surrogate Modeling toolbox

Set-up of the platform for surrogate generation is given in Figure 2.15. A spread layout of task execution is shown in Figure 2.16.

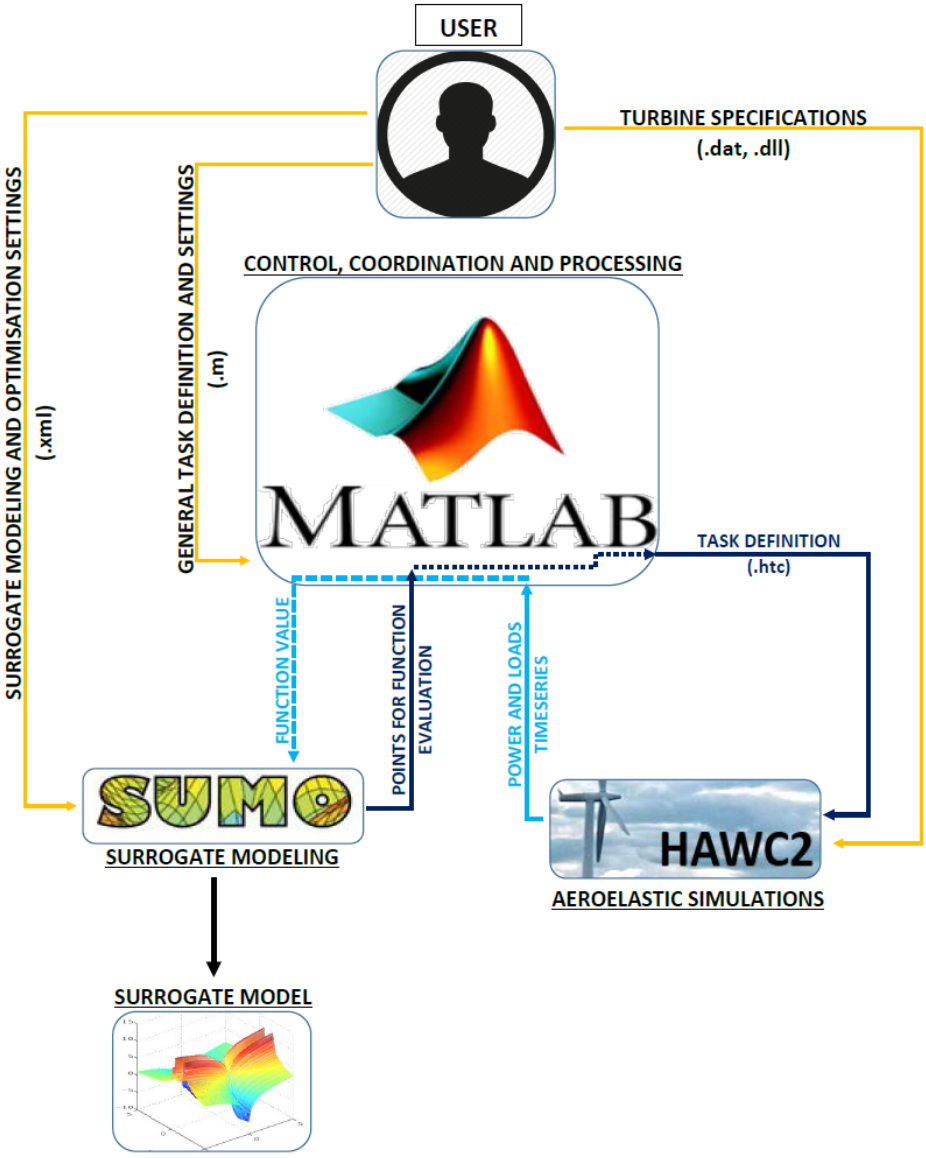


Figure 2.15: Platform set-up for surrogate modeling

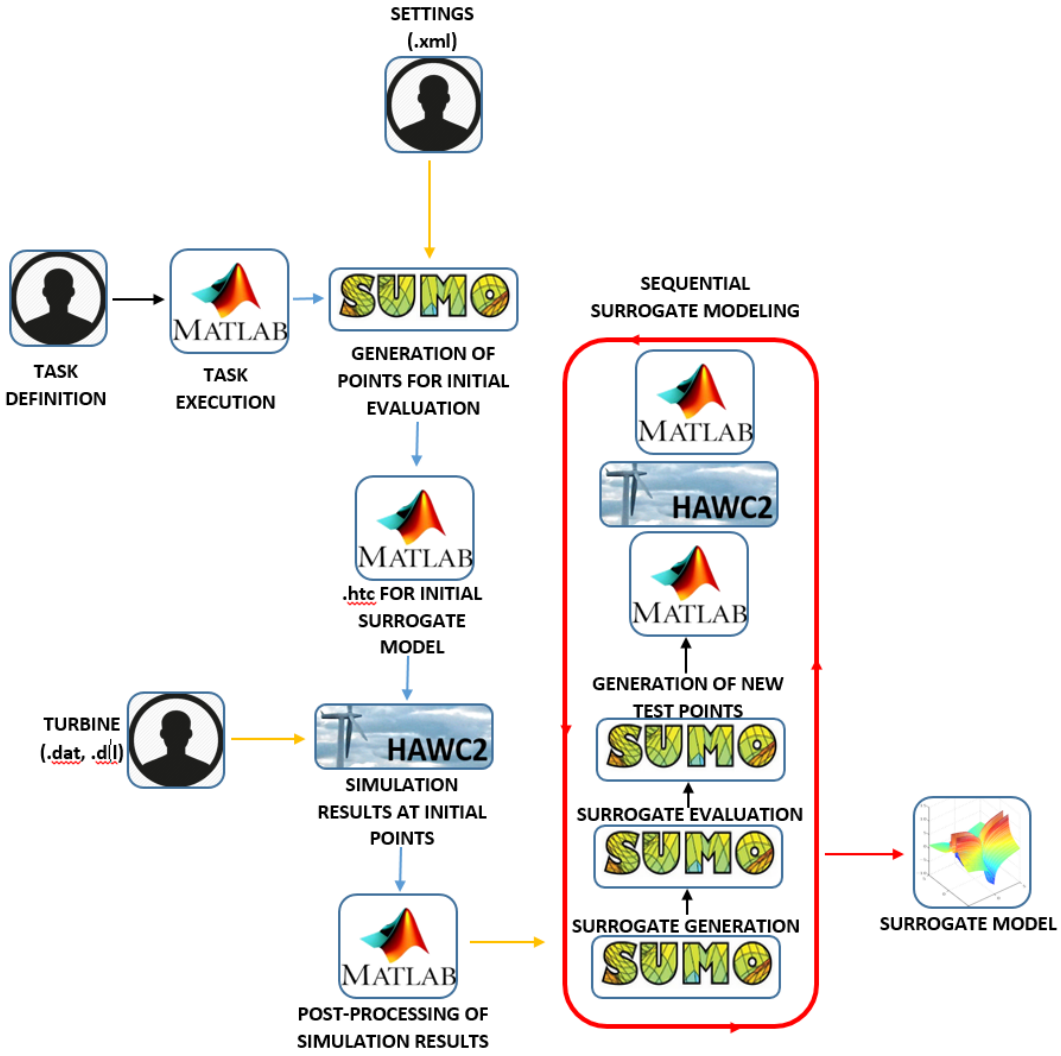


Figure 2.16: *Platform set-up for surrogate modeling. Spread layout*

The main directory, from where the coordination is performed, is the `scratch` folder on the cluster. This is thus the location where all toolboxes, scripts relevant for surrogate modeling and input files are stored. All simulations results will also be copied to a designated folder within the `scratch` directory.

Modeling of a surrogate is started by running a SUMO toolbox through provided MATLAB script. Prior to the run, several input files need to be supplied and configured.

Input files required for the HAWC2 simulations are the definition of turbine components, their structural and aerodynamic properties and the controller. In this project, files for NREL 5MW turbine are used in their default state.

A **config.xml** is the main file, where the settings for the surrogate model are specified. (Settings include those discussed in section 2.2). Within the **config.xml** file, one also

needs to define the **simulator**, i.e. how the data points are gathered. This is done by pointing to **simulator.xml** file, which contains the definition of inputs and outputs of the surrogate model as well as path to the **simulator** function and/or sets of already existing samples.
Aeroelastic simulations of wind turbine are performed by HAWC2. The results are then post-processed by a collection of MATLAB scripts and functions. In order to establish the interaction between SUMO and HAWC2, another set of MATLAB scripts is used. Thus, the path to the simulator, in **simulator.xml**, points to MATLAB script, which receives sampling locations determined by SUMO, creates **.htc** input files for HAWC2, and for each **.htc** file, it creates a **.p** file. A **.p** file is a script specifying settings for running HAWC2 simulations on a cluster. Most important ones are:

- definition of the files that need to be copied to a cluster node
- copying of these files to a node
- command that initiates the run of simulations on a node
- definition of files that need to be copied from the node back to the main directory
- copying of these files to the main directory

Distribution of tasks (**.p** files) to nods and initization of these is handled by the **launch.py** script stored in the cluster repository. **launch.py** was prepared by David Verelst. Same author also provided the template for the **.p** files. In order to run **launch.py** on the cluster, X-emulator and a shell client are required. X-emulator is automatically handled on all DTU computers, and the common choice of the shell client is PuTTY. In order to ensure continuous and uninterrupted flow of information, running of external software (in this case PuTTY) is done through **command prompt** called from MATLAB. However, logging onto the cluster requires user name, password, the name of the script which needs to be ran (**launch.py**) and number of required CPUs. Thus, the software which is ran from the **command prompt** has to have all of these inputs defined as `command line attributes`. Since PuTTY does not support all of these, an alternative programme KiTTY is used.
During the period from when MATLAB initiates KiTTY to the point when all the simulation files are copied back to the main directory, a monitoring function is employed. This was also written as a MATLAB function, and its purpose is to pause modeling until all results are made available. This function is a requirement, since SUMO expects results directly from specified simulator file and does not know whether additional programs are

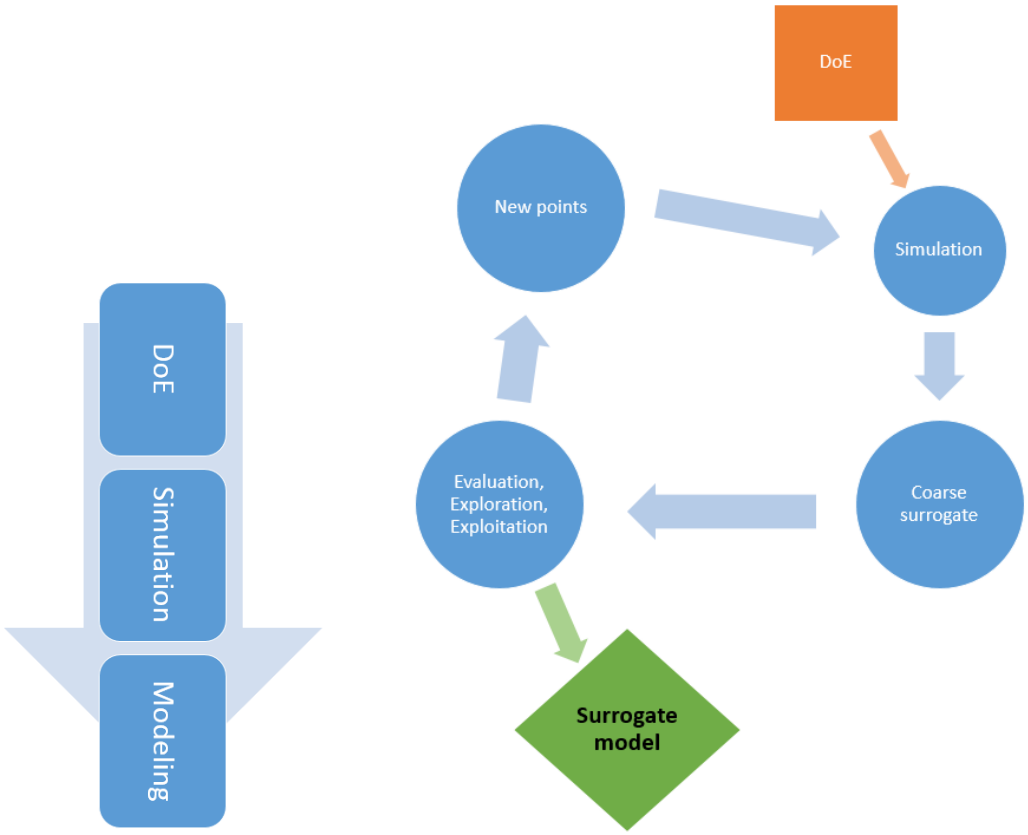
used in-between.

Once the simulations on the cluster are completed, result files (**.sel**, **.res**, **.log**) are copied to the main directory, monitoring function signals about their "existence", and the post-processing of the data is initiated by a collection of MATLAB scripts. These have been provided by DTU wind energy department and were adjusted only to specify particular sensors for data gathering and to define naming of the files used throughout the post-processing. The time series signal of loading at different location on a turbine is processed using rain-flow counting (existing MATLAB function). Result is the lifetime equivalent loading at each sensor which, together with the power output, is supplied back to SUMO as a data point.

Surrogate modeling is further handled by SUMO, and when samples are needed, simulator is called again. The modeling is performed until one of the stopping criteria specified in **config.m** is fulfilled. These may be maximum number of samples, maximum time allowed for optimisation of hyper-parameters, maximum number of iterations or targeted error tolerances.

2.3.2 One-shot coarse surrogate model

"One-shot design", as opposed to "sequential design", indicates only samples available prior to the modeling phase are used (see Figure 2.17). The modeling in this case refers to hyper-parameter optimisation). In this, 1*st* stage, phase, a low cost, so-called, "one-shot coarse" surrogate model is created. In addition to the verification of the set-up, this model is used to gather knowledge about global trends in the behaviour as well as to test which type of the surrogate is most appropriate for each output. Based on this information, settings for creating more optimal surrogate will be determined. Furthermore, samples gathered in the this stage will be used as a validation set for the sequential modeling.

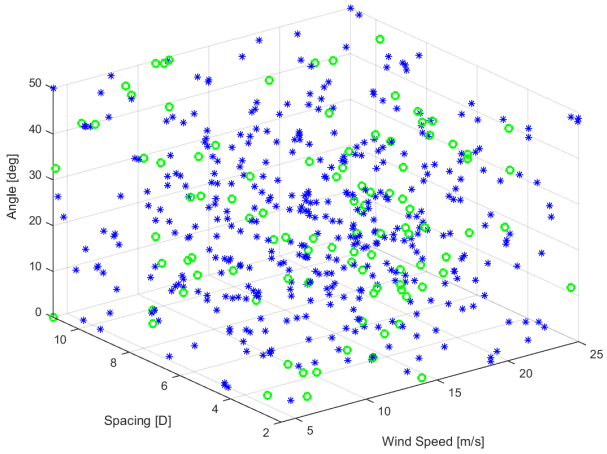


(a) One-shot modeling (b) Sequential modeling

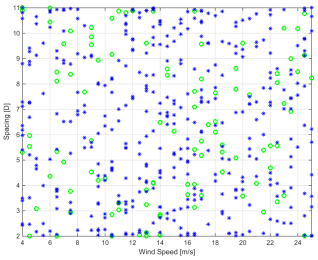
Figure 2.17: *Surrogate modeling approaches*

In the 1*st* stage, the HAWC2 DWM model simulations were ran in 500 random locations within the design space. These are shown in Figure 2.18. Wind speed was sampled in the range from 4m/s (cut-in wind speed) to 25m/s (cut-out wind speed), spacing was sampled from 2D to 11D, and the wake angle from 0° to 50°. Note that in order to decrease the

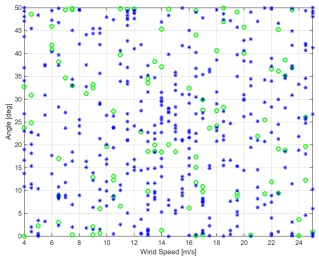
number of turbulent files, wind speed was rounded to precision of 0.5.
In order to use the sampling error estimation, 500 samples are split in two sets, the training set featuring 400 points, and the validation set featuring 100 points.



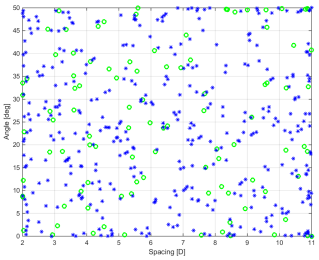
(a) 3D design space



(b) projection 1



(c) projection 2



(d) projection 3

Figure 2.18: *Initial set of 500 randomly chosen samples. Training points are marked as blue ∗, validation points are marked as green o.*

Following types of surrogate were used: Kriging interpolation, rational function model (polynomia regression), radial basis functions, and a heterogenetic model. The heterogentic algorithm builds surrogates using several model types and propagates the bests by using a genetic algorithm.
The objective of the modeling is defined as minimisation of the error. The measures are used are the 5-fold cross-validation using Average Euclidean error function and the Root Relative Mean Square error, which is default setting for the validation set.
The statistics for each model is given in Figure 2.19 and Figure 2.20. The legend of labels is given bellow.

ID	label
Equivalent load (Mx) at the blade root	baldeMx
Equivalent load (My) at the blade root	bladeMy
Equivalent load (Mx) at the tower bottom	towerMx
Equivalent load (My) at the tower bottom	towerMy
Equivalent load (Mx) at the shaft	shaftMx
Equivalent load (Mz) at the shaft	shaftMy

WFM vSM kriging 06 26					Mean relative error
Kriging					
One-SHOT					
number of training samples: 400					
number of validation samples: 100					
Individual Measures					
Average model error (considering all measures):	36%				
Average model error (considering only AEE & RRSE):	22%				
	targets reached: 14%				237%
POWER : Cross Validation - average Euclidean Error	0,01	0,04	-65%	1	-53%
POWER: Cross Validation - Mean Absolute Error off	0,11	0,05	117%		
POWER: Validation Set - Root Relative Square Error	0,06	0,10	-41%	1	
bladeMx: Cross Validation - Mean Absolute Error off	0,44	0,05	786%		555%
bladeMx: Cross Validation - Root Relative Square Error	0,26	0,04	552%	0	
bladeMx: Validation Set - Root Relative Square Error	0,26	0,04	558%	0	
bladeMy: Cross Validation - Mean Absolute Error off	0,08	0,05	67%		22%
bladeMy Cross Validation - Root Relative Square Error	0,18	0,15	21%	0	
bladeMy: Validation Set - Root Relative Square Error	0,18	0,15	23%	0	
shaftMx: Cross Validation - Mean Absolute Error off	0,23	0,05	361%		279%
shaftMx: Cross Validation - Root Relative Square Error	0,16	0,04	302%	0	
shaftMx: Validation Set - Root Relative Square Error	0,14	0,04	256%	0	
shaftMz: Cross Validation - Mean Absolute Error off	0,99	0,05	1886%		317%
shaftMz: Cross Validation - Root Relative Square Error	0,27	0,05	449%	0	
shaftMz: Validation Set - Root Relative Square Error	0,26	0,09	184%	0	
towerMx: Cross Validation - Mean Absolute Error off	0,62	0,05	1148%		397%
towerMx: Cross Validation - Root Relative Square Error	0,27	0,05	439%	0	
towerMx: Validation Set - Root Relative Square Error	0,23	0,05	354%	0	
towerMy: Cross Validation - Mean Absolute Error off	2,03	0,05	3953%		140%
towerMy: Cross Validation - Root Relative Square Error	0,31	0,25	26%	0	
towerMy: Validation Set - Root Relative Square Error	0,42	0,12	254%	0	

(a) Kriging

WFM vSM rational 06 26					Mean relative error
RATIONAL					
One-SHOT					
number of training samples: 400					
number of validation samples: 100					
Individual Measures					
Average model error (considering all measures):	32%				
Average model error (considering only AEE & RRSE):	20%				
	targets reached: 7%				246%
POWER : Cross Validation - - average Euclidean Error	0,04	0,04	6%	0	9%
POWER:Cross Validation - Mean Absolute Error off	0,21	0,05	324%		
POWER:Validation Set - Root Relative Square Error	0,11	0,10	11%	0	
bladeMx: Cross Validation - Mean Absolute Error off	0,36	0,05	619%		377%
bladeMx: Cross Validation - Root Relative Square Error	0,21	0,04	419%	0	
bladeMx: Validation Set - Root Relative Square Error	0,17	0,04	336%	0	
bladeMy Cross Validation - Mean Absolute Error off	0,05	0,05	4%		2%
bladeMy Cross Validation - Root Relative Square Error	0,16	0,15	6%	0	
bladeMy Validation Set - Root Relative Square Error	0,15	0,15	-3%	1	
shaftMx Cross Validation - Mean Absolute Error off	0,44	0,05	777%		602%
shaftMx Cross Validation - Root Relative Square Error	0,30	0,04	646%	0	
shaftMx Validation Set - Root Relative Square Error	0,26	0,04	557%	0	
shaftMz: Cross Validation - Mean Absolute Error off	0,61	0,05	1119%		134%
shaftMz: Cross Validation - Root Relative Square Error	0,16	0,05	211%	0	
shaftMz: Validation Set - Root Relative Square Error	0,14	0,09	56%	0	
towerMx Cross Validation - Mean Absolute Error off	0,58	0,05	1054%		553%
towerMx Cross Validation - Root Relative Square Error	0,34	0,05	578%	0	
towerMx Validation Set - Root Relative Square Error	0,31	0,05	528%	0	
towerMy Cross Validation - Mean Absolute Error off	1,61	0,05	3111%		46%
towerMy Cross Validation - Root Relative Square Error	0,25	0,25	1%	0	
towerMy Validation Set - Root Relative Square Error	0,23	0,12	91%	0	

(b) Rational

Figure 2.19: *Statistics for building one-shot model*

WFM vSM rbf26 06					Mean relative error
RBF					
One-SHOT					
number of training samples: 400					
number of validation samples: 100					
Individual Measures					
Average model error (considering all measures):	25%				
Average model error (considering only AEE & RRSE):	13%				
	targets reached: 36%				99%
POWER : Cross Validation - - average Euclidean Error	0,02	0,04	-61%	1	
POWER:Cross Validation - Mean Absolute Error off	0,14	0,05	189%		-57%
POWER:Validation Set - Root Relative Square Error	0,05	0,10	-53%	1	
bladeMx: Cross Validation - Mean Absolute Error off	0,18	0,05	262%		
bladeMx: Cross Validation - Root Relative Square Error	0,14	0,04	255%	0	233%
bladeMx: Validation Set - Root Relative Square Error	0,12	0,04	211%	0	
bladeMy Cross Validation - Mean Absolute Error off	0,05	0,05	-10%		
bladeMy Cross Validation - Root Relative Square Error	0,12	0,15	-20%	1	-27%
bladeMy Validation Set - Root Relative Square Error	0,10	0,15	-34%	1	
shaftMx Cross Validation - Mean Absolute Error off	0,17	0,05	242%		
shaftMx Cross Validation - Root Relative Square Error	0,10	0,04	154%	0	138%
shaftMx Validation Set - Root Relative Square Error	0,09	0,04	121%	0	
shaftMz: Cross Validation - Mean Absolute Error off	0,52	0,05	934%		
shaftMz: Cross Validation - Root Relative Square Error	0,12	0,05	136%	0	88%
shaftMz: Validation Set - Root Relative Square Error	0,13	0,09	40%	0	
towerMx Cross Validation - Mean Absolute Error off	0,44	0,05	776%		
towerMx Cross Validation - Root Relative Square Error	0,20	0,05	299%	0	284%
towerMx Validation Set - Root Relative Square Error	0,18	0,05	268%	0	
towerMy Cross Validation - Mean Absolute Error off	1,85	0,05	3595%		
towerMy Cross Validation - Root Relative Square Error	0,23	0,25	-8%	1	33%
towerMy Validation Set - Root Relative Square Error	0,21	0,12	74%	0	

(a) Radial basis functions

WFM vSM heterogenetic 06 26					Mean relative error
HETEROGENETIC					
One-SHOT					
number of training samples: 400					
number of validation samples: 100					
Individual Measures					
Average model error (considering all measures):	24%				
Average model error (considering only AEE & RRSE):	13%				
	targets reached: 36%				96%
POWER : Cross Validation - - average Euclidean Error	0,02	0,04	-58%	1	
POWER:Cross Validation - Mean Absolute Error off	0,16	0,05	212%		-55%
POWER:Validation Set - Root Relative Square Error	0,05	0,10	-52%	1	
bladeMx Cross Validation - Mean Absolute Error off	0,18	0,05	269%		
bladeMx Cross Validation - Root Relative Square Error	0,14	0,04	250%	0	220%
bladeMx Validation Set - Root Relative Square Error	0,12	0,04	191%	0	
bladeMy Cross Validation - Mean Absolute Error off	0,05	0,05	-5%		
bladeMy Cross Validation - Root Relative Square Error	0,12	0,15	-18%	1	-27%
bladeMy Validation Set - Root Relative Square Error	0,09	0,15	-37%	1	
shaftMx Cross Validation - Mean Absolute Error off	0,18	0,05	269%		
shaftMx Cross Validation - Root Relative Square Error	0,10	0,04	158%	0	140%
shaftMx Validation Set - Root Relative Square Error	0,09	0,04	122%	0	
shaftMz: Cross Validation - Mean Absolute Error off	0,53	0,05	964%		
shaftMz: Cross Validation - Root Relative Square Error	0,11	0,05	122%	0	77%
shaftMz: Validation Set - Root Relative Square Error	0,12	0,09	33%	0	
towerMx Cross Validation - Mean Absolute Error off	0,44	0,05	770%		
towerMx Cross Validation - Root Relative Square Error	0,21	0,05	318%	0	287%
towerMx Validation Set - Root Relative Square Error	0,18	0,05	255%	0	
towerMy Cross Validation - Mean Absolute Error off	1,78	0,05	3452%		
towerMy Cross Validation - Root Relative Square Error	0,23	0,25	-7%	1	34%
towerMy Validation Set - Root Relative Square Error	0,21	0,12	74%	0	

(b) Heterogenetic

Figure 2.20: *Statistics for building one-shot model*

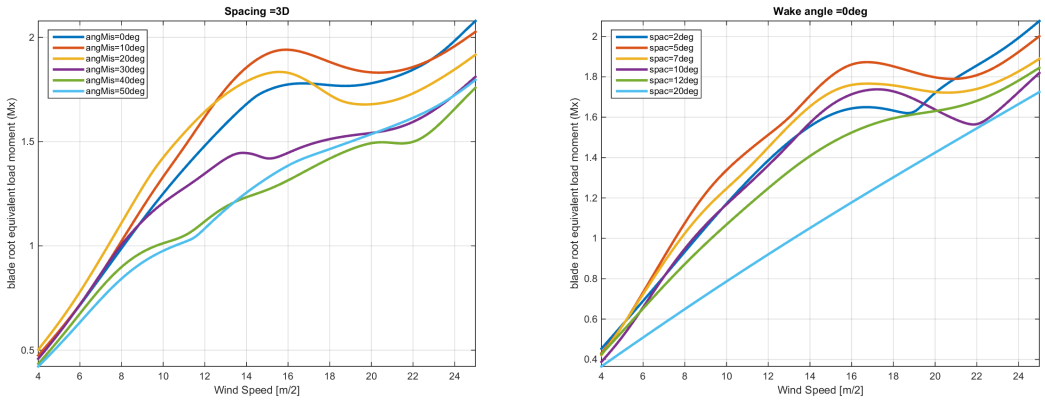
Heterogenetic modeling resulted in radial basis functions for each output except the output `towerMx` which was modelled as an ensemble of several models. This is consistent when compared to the results of different model types, where RBF resulted in optimal values for each output. Nevertheless, considering also the MAE grades, rational model for "towerMy" and Kriging for "POWER" are slightly better then the RBF. Notice, however, only `POWER` and `bladeMy` outputs satisfied set targets. Furthermore, measuring output `POWER` by MAE, one can notice it exceeds the targeted value by 117% using Kriging and by 189% using RBF. Note this is the relative offset from the specified target and not the value of the measure. Moreover, the data modeled by the surrogate is normalized. Thus this measure needs to be returned in its original range, if one should analyse it further.
Error measures of the "one-shot coarse surrogate model" in original range (using validation set of 100 points) are shown in Table 2.3. Note again, some measures are not straightforward to interpret, and should best be be compared to set targets.
In addition to these measures, one may also want to check the cross-validation measure in Figure 2.20 as a way of interpreting how sensitive the model is to the points used for its creation.

Table 2.3: *Measures of "one-shot coarse surrogate model". Equivalent loads are given in [kNm] and power output in [kW].*

Measure	bladeMx	bladeMy	towerMx	towerMy	shaftMx	shaftMz	Power
maximal sample value	11580.29	8478.88	24391.67	18327.13	8542.49	1292.89	5000.22
maximal surrogate value	11627.01	8472.10	23380.62	18745.69	8416.54	1279.10	5139.32
minimal sample value	2295.66	5817.67	8827.83	2145.00	2830.57	74.57	172.76
minimal surrogate value	2289.29	5824.29	8695.64	2332.65	2786.03	66.24	100.93
Average Euclidean Error	198.31	43.04	443.08	631.12	66.96	20.47	52.86
Root Relative Square Error	0.12	0.09	0.17	0.21	0.09	0.12	0.05
Maximal Absolute Error	879.39	223.98	2933.58	2514.05	624.24	258.61	415.77
Mean Absolute Error	198.31	43.04	443.08	631.12	66.96	20.47	52.86
Mean Relative Error	0.03	0.01	0.03	0.11	0.01	0.04	0.04
Maximal Relative Error	0.15	0.03	0.2	0.6	0.12	0.60	0.5

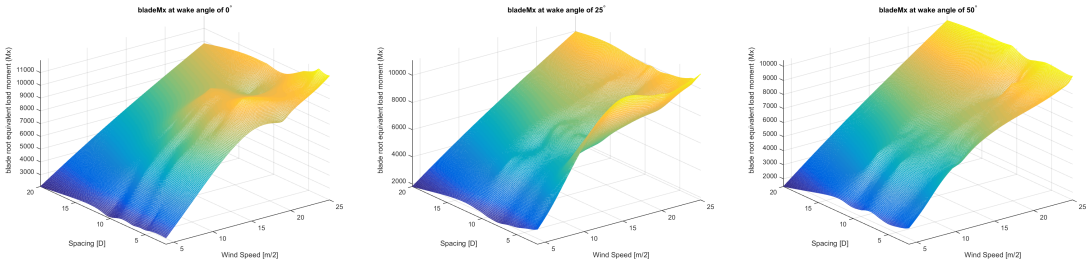
The importance of measuring the surrogate with different error functions is clearly visible from Table 2.3. Notice while one may say the surrogate has a relative error in the order of 5% (except output `towerMy`) which is reasonable, maximum errors show the offset to be higher in some regions of the design space. It is expected that the sequential design will take care of these. Notice further, the outputs `towerMy`, `shaftMz` and `Power` show the largest error, which is an indication that these will be most difficult to fit.

It is rather hard to make a good representation of a 4-dimansional model (3 inputs, 1 output). Example of 2D plots generated at various sections are shown Figure 2.21 and Figure 2.22 for output `bladeMx`. Documentation providing the quality of the developed surrogate model (any output) as well as extensive 2D and surface plots is generated by a code supplied with the thesis. Example of such a report sheet is given in section B.1.



(a) Spacing is 3D (b) Wake angle is 0°

Figure 2.21: *Results of one-shot design for lifetime fatigue equivalent load bending moment at the blade root in falpwise direction.*



(a) Wake angle is 0° (b) Wake angle is 25° (c) Wake angle is 50°

Figure 2.22: *Results of one-shot design for lifetime fatigue equivalent load bending moment at the blade root in falpwise direction*

2.3.3 What can be learnt from the coarse surrogate model?

Although the sampling points for the coarse surrogate were chosen randomly on purpose, no particular reasoning about the range nor the rounding of each input was made. Even more, the surrogate was created without any regard to its further use and the allowed error was chosen arbitrarily. Nevertheless, as stated in the beginning of previous section, the aim of the coarse surrogate is to learn about the behaviour of outputs so the settings for creating the surrogate through sequential design may be improved.
Main question addressed while creating the surrogate for the purpose of WFLO is how precise should it be. By answering this, one should have a better idea of how many simulations are required to build an optimal model. The required precision of the surrogate model depends however on objective function for which the surrogate will provide input. Since the platform for wind farm analysis is not yet ready at this stage, the precision required from the surrogate will be determined by the convergence of the modeling process and by the constraint for the maximal number of samples.
The accuracy of the surrogate model is examined through sensitivity analysis, which will also serve as a way of determining the rounding of the surrogate design variables (wind speed, spacing and wake angle).

Point of view: sensitivity analysis

According to [38], a first order approach to analyse the sensitivity of the model to its inputs is to examine the scatter plots for input versus the model value. A trend or a pattern in the data is an indication how influential the variable is.
Example in Figure 2.23 is given for output `towerMx`. Notice the visible trend in the points, especially when looking at the wake angle versus the loading values. Dependency on the wind speed is also expected. Notice from Figure 2.23, the surrogate model varies over entire range of wind speeds, while for the wake angle only until $\approx 20°$. Surprisingly, the spacing shows little effect.
A more precise measure on sensitivity is to estimate the gradient and hessian, however, even for the surrogate this might be too expensive for this purpose. A measure of the total sensitivity effect using conditional variance of the samples given in Equation 2.11 [38] is used instead. Note this is a measure of sensitivity to input variables and not measure of

uncertainty.

$$S_{tot,i} = 1 - \frac{V(E(Y|X_{\ i}))}{V(Y)} \tag{2.11}$$

where

$S_{tot,i}$ is the total sensitivity of variable i

$V(...)$ is the variance operator

$E(...)$ is the mean operator

Y are the output values

$Y|X_{\ i}$ means operation is carried over output Y for all inputs fixed except $X_{\ i}$

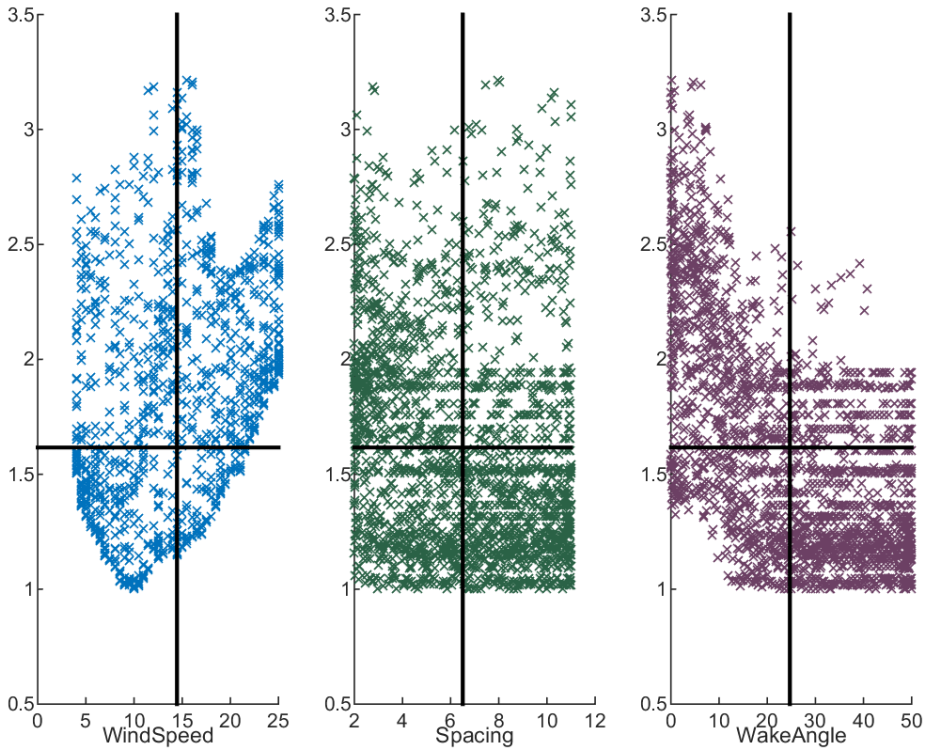


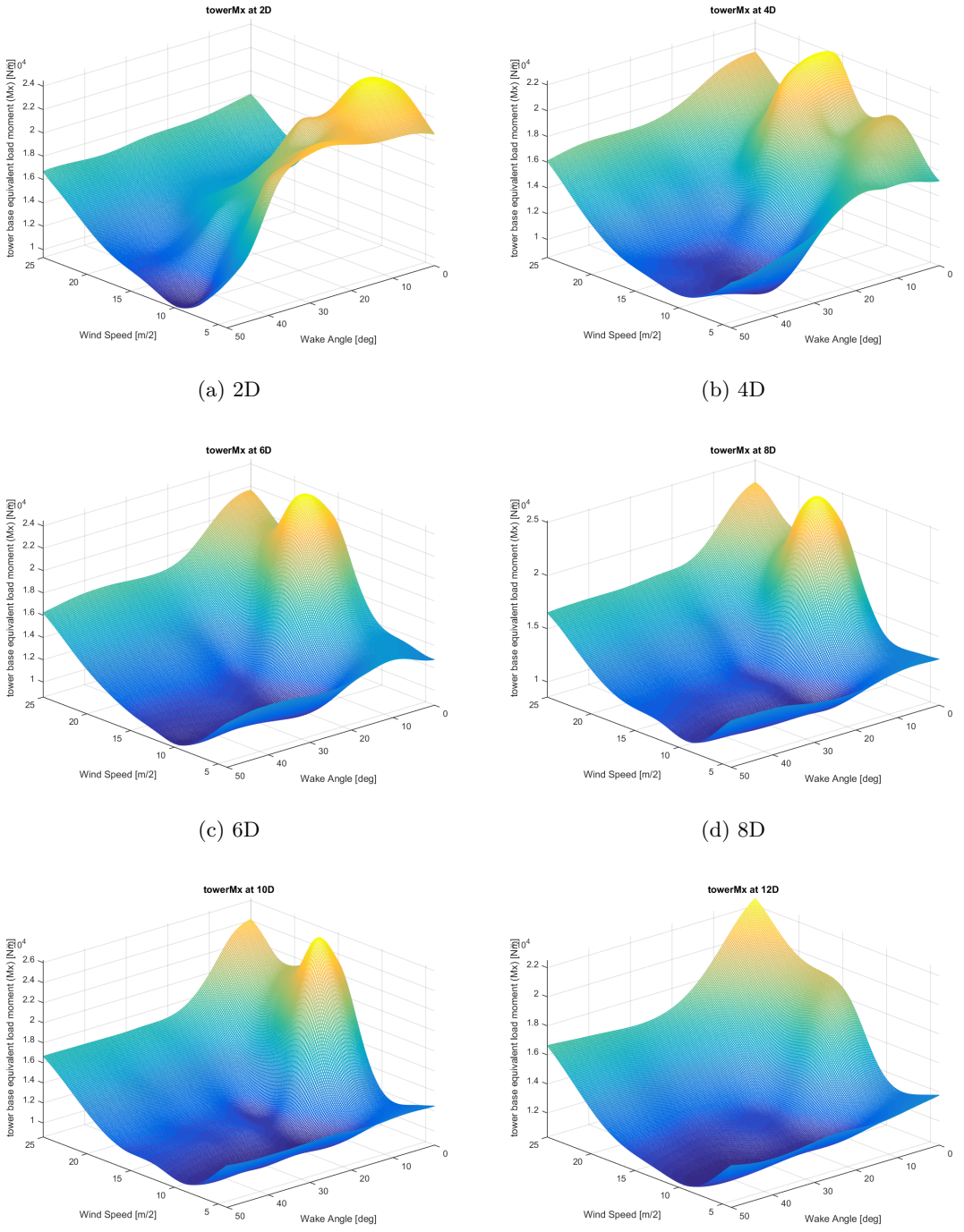
Figure 2.23: *Scatter plot of samples used to create one-shot coarse surrogate model. Output:* `towerMx`

Results for sensitivity analysis of the coarse surrogate model using Equation 2.11 are given in Table 2.4. The dependency on the wind speed is highest. Spacing and wake angle have more or less equal effect to the model. An exception is the output `towerMx`. According to the sensitivity analysis, wake angle is confirmed to have the highest influence on the

loading of the tower in front-aft movement. A confirmation is also made by inspecting the 3D surface plot in Figure 2.24. It is clearly seen the loading changes more abruptly w.r.t. the wake angle compared to the wind speed.

Table 2.4: *Sensitivity analysis of the coarse surrogate model to the inputs*

output	WindSpeed	Spacing	WakeAngle
'bladeMx'	0.896	0.062	0.114
'bladeMy'	0.93	0.07	0.08
'towerMx'	0.49	0.41	0.69
'towerMy'	0.95	0.09	0.10
'shaftMx'	0.83	0.28	0.25
'shaftMz'	0.99	0.01	0.01
'Power'	0.97	0.04	0.09



(a) 2D (b) 4D

(c) 6D (d) 8D

(e) 10D (f) 12D

Figure 2.24: *Coarse surrogate model of output* `towerMx` *at different wind turbine spacing.*

While data in Table 2.4 says on which variable surrogate depends the most, it does not give any suggestion which precision should input variables be rounded to. Determining

the levels each variable can be sampled on is important because it gives a good estimate of maximal number of points required for let's say full-factorial design. A model built with less amount of points may then be called an improvement.

To test the sensitivity to rounding, a change in output value was analysed for various step sizes.

An example for sensitivity to change in wind speed (in this case labelled as x_1) is given in Equation 2.12. Note that δ in Equation 2.12 is computed on a 2D grid discretized in 300 points and that the values are averaged over wind speed. One should realise this approach is not entirely valid, since the behaviour of the surrogate varies in different regions of its domain. The method used in this section is presented simply as a first order approach to determine the number of levels for each variable with the aim of restricting the computational cost.

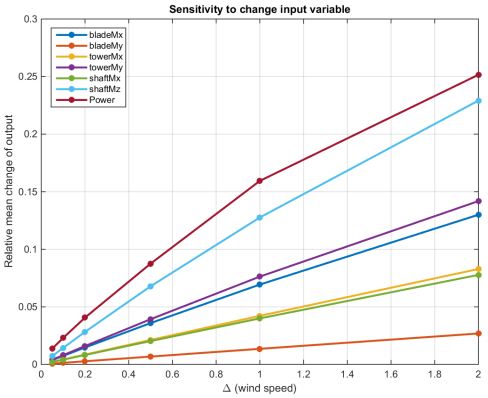
Results for sensitivity to rounding of input variables are given in Figure 2.25, values of mean and standard deviation are further listed in Table 2.5 and Table 2.6, respectively. Conclusion is that if we wish to capture changes in the order of 5% (averaged over design space of each model), wind speeds should be sampled at least every 0.2m/s, spacings every 0.5D and wake angles every 2°. For the same range as used in creation of the one-shot coarse model, the full factorial design would require over 50 000 points.

$$\delta = \frac{|f(\chi) - f(\chi + \Delta x)|}{f(\chi)} \tag{2.12}$$

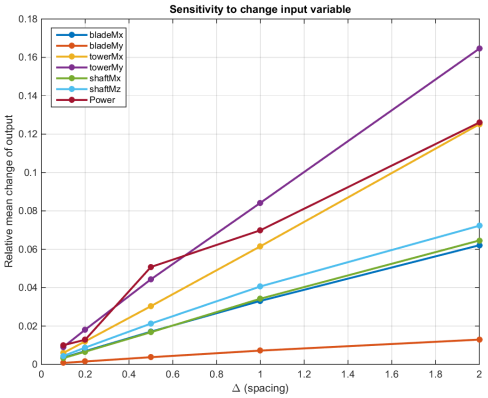
where

$\chi = [x_1, x_2, x_3]$

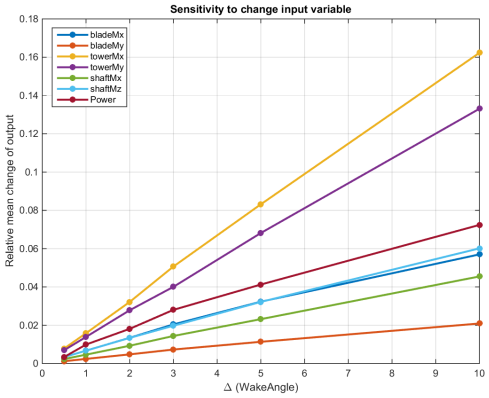
$\Delta x = [\Delta x_1, 0, 0]$ is a vector with non-zero element on a position of input being analysed



(a) w.r.t. wind speed



(b) w.r.t. spacing



(c) w.r.t. wake angle

Figure 2.25: *Sensitivity of surrogate to change in input*

Table 2.5: *Mean relative sensitivity to inputs*

rounding	bladeMx	bladeMy	towerMx	towerMy	shaftMx	shaftMz	Power
0.10	0.01	0.00	0.00	0.01	0.00	0.01	0.0230
0.20	0.01	0.00	0.01	0.02	0.01	0.03	0.0406
0.50	0.04	0.01	0.02	0.04	0.02	0.07	0.0875
1.00	0.07	0.01	0.04	0.08	0.04	0.13	0.1594

Table 2.6: *Standard deviation of relative sensitivity to inputs*

rounding	bladeMx	bladeMy	towerMx	towerMy	shaftMx	shaftMz	Power
0.1000	0.0004	0.0001	0.0005	0.0011	0.0006	0.0014	1.0157
0.2000	0.0008	0.0002	0.0011	0.0021	0.0011	0.0026	1.1802
0.5000	0.0018	0.0004	0.0028	0.0052	0.0027	0.0058	0.6373
1.0000	0.0033	0.0008	0.0059	0.0100	0.0051	0.0098	0.7795

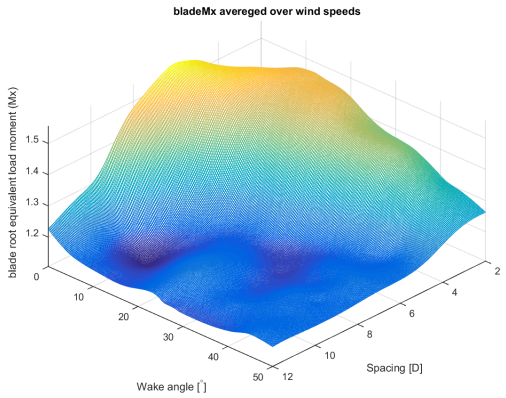
Apart from the sensitivity of the surrogate itself, it is important to check weather other modules involved in the simulation are able to provide results of demanded accuracy. A test inspecting this was performed on the HAWC2 aeroelastic tool. The aim was to see how accurate the mean wind speed of the turbulence box is w.r.t. the mean wind speed set in the definition (**.htc**) file. Results of this analysis show the rounding error to be in the order of $\epsilon = 0.025$ with the tendency to decrease towards higher wind speeds.

Point of view: layout evaluation

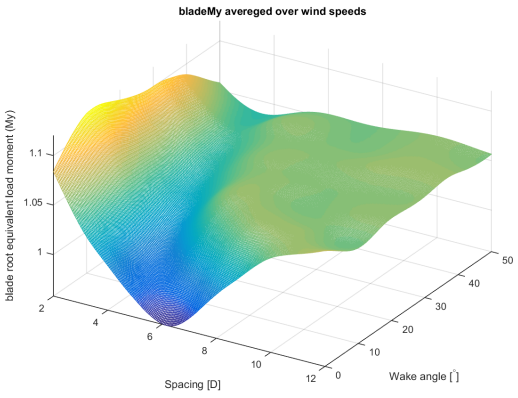
Going back to the evaluation of wind farm layouts one may notice each wind turbine is sdimulated for each discrete wind speeds from cut-in of 4m/s to cut-out of 25m/s in steps of 1m/s. Furthermore, analysis of production is performed for wind coming from 12 directions (from 0° to 330° in steps of 30°). Having this in mind, it can be noticed:

1. Samples taken outside the wind speed values at which the turbine is simulated serve only for fitting.

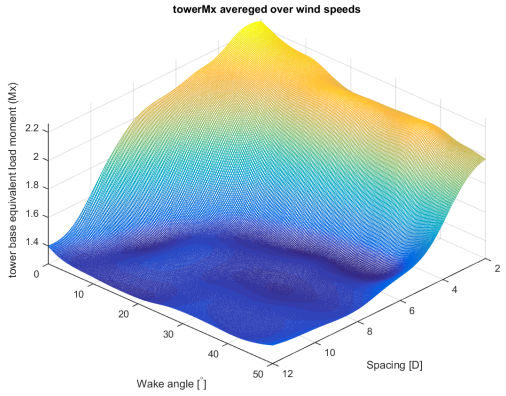
2. Each of the 12 sectors covers an angle of 30° (±15°) which means wake angles over 15° will not occur. As a consequence, the surrogate model may be modeled using the wake angle range from 0° to 15°, which significantly reduces the computational effort. However, due to meandering there might be cases when the wake angles above 15° cause high loading. As it will be discussed further in section 4.1, the "12 30°-wide" sector analysis does not seem as an optimal solution.
   An analysis of the outputs was made to check when the highest loads occur. Surface plots of loading averaged over wind speed are shown in Figure 2.26. Note, however, these models have to be further scaled with the probability of each wind speed bin. Thus, the locations of highest loads will shift when a particular site comes into consideration. However, observing the averaged loads, it can be noticed that the increase of wake angle has a tendency to reduces the loads while the highest loading occurs in full-wake cases or for small wake angles. The exception is the output **shaftMz**, however, for this output the model showed poor fitting results.



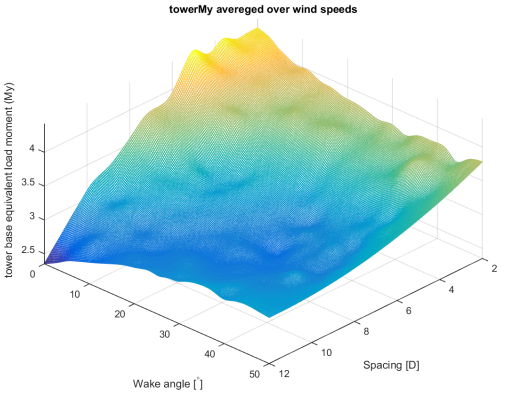
(a) Normalised loading output:`bladeMx`



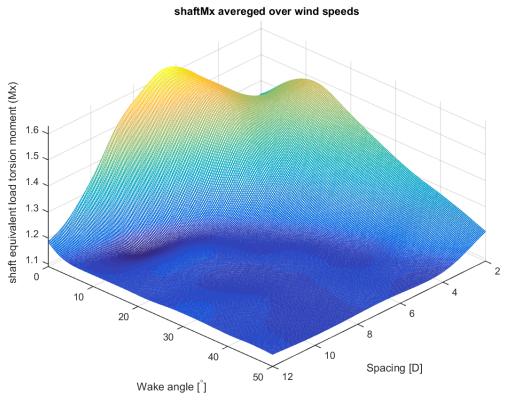
(b) Normalised loading output:`bladeMy`



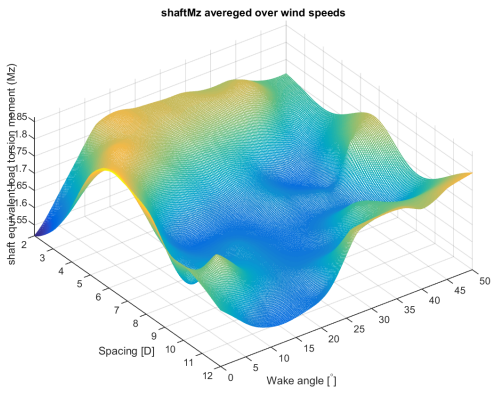
(c) Normalised loading output:`towerMx`



(d) Normalised loading output:`towerMy`



(e) Normalised loading output:`shaftMx`



(f) Normalised loading output:`shaftMz`

Figure 2.26: *Loading at different sensors averaged over wind speeds of equal probability*

The version of DWM model used for this project is not able to consider effects of multiple wakes, and only the turbine causing highest loading is taken to be the significant one for

that particular wind direction. It is decided the wake angle should be sampled in the range from 0° to 30° in which case for each wind direction a sector covers a width of 60°.

In the conclusion, one-shot design coarse surrogate model has been built with the aim of capturing global behaviour of each output. A further goal was to test different type of surrogate models and understand the need for multi-objective measuring. The one-shot surrogate was created using a training set of 400 samples and validation set of 100 samples. The samples have been chosen randomly, however to decrease the computational cost of creating turbulence boxes, sampling of the wind speed has been restricted to precision of 0.5m/s. The resulting model has a mean relative error of $\mu = 0.038$ with a standard deviation of the $\sigma = 0.035$.
Based on the conclusion derived from the coarse surrogate model, the continuation of the chapter will present a plan to build a surrogate of higher fidelity.

2.3.4 Sequential design surrogate

Settings

From the analysis of the coarse surrogate model following settings are chosen:

- Sampling of the wind speed should be at least every 0.2m/s for the range between 4m/s and 25m/s. Knowing that each wind speed value requires 9 turbulent files, such sampling will require 954 files. Creation of these brings additional computational effort as well as memory requirement. However, note that once these files are created, they are stored on the cluster and may be used repeatedly. Thus, effort required considering the turbulent files is of one-time nature only.

- Rounding of spacing and wake angles is done with the purpose of reducing the number of possible samples. From the sensitivity analysis, spacing should be sampled at least every 0.5D in the range from 2D to 15D and wake angles every 2° in the range from 0° to 30°.

- Even though the changes in surrogate behaviour should be captured with sampling presented above, one may argue the fitting of the model depends more on the specific position of the sampling point and less on the number of levels. While that may as well be the fact, a compromise must be reached somewhere.

- Upper limit of total number of point was set to be 1000 (for training)+421(for validation). This is approximately 2% of the number of points required for the full-factorial design.

- A cluster was made available for the purpose of running larger number of simulations in parallel. Nevertheless, in order to take maximal advantage of the sequential design and use the knowledge gathered in each step, the number of simulations per run is constrained to 150. In case where this would not be done, the algorithm may perform total amount of allowed simulations in "1-go". Such a model would have equal performance as the "one-shot" design.
  In practice, approximately 50 simulations per step were ran in parallel. The exception is the initial design which was defined by LHS of 400 points and 21 additional points at the domain edges. As a consequence of limiting the number of simulations per step, time required for creating the model is increased compared to the case where all points could be evaluated in 2-3 cluster runs. In this project, however, higher consideration was put on ensuring the sampling is done in a smart, gradual way and less on the speed of computations.

- Based on the grading of the coarse surrogate, RBF type model was assigned to all outputs with the exception of **towerMy** and **shaftMz**. Because of poor fitting results compared to others, these outputs were modeled by heterogenetic model in order to reach the ensemble model as a result. Note that the use of the heterogentic modeling does not require extra samples and the computational effort should not increase much. Heterogenetic model will build surrogate models using several surrogate types and propagate the best individuals using genetic algorithm. This means it will act as a global search tool, and will not be restrained by one modeling type. Furthermore, heterogenetic modeling setting allows the use of ensemble model, which combines the use of different model types over different portions of the domain.

- The method for determining locations for new samples was left on "default" setting, which means 70% of the points are chosen using the LOLA-Voronoi algorithm, and the rest is sampled in the location where the surrogate shows largest offset from the validation set.

- No restriction on the run-time has been set.

- Measures used during the modeling are the 5-fold cross-correlation using the Mean Relative Error function and Root Relative Square Error on the validation set.

- Objective was adjusted to approach the mean relative error of 5% considering validation set. The cross-correlation targets are listed for each output in Figure 2.27.

Once the maximum allowable number of simulations has been performed, SUMO terminates the modeling process. For this reason, the gathered data is used to additionally tune-up the model. This is done in a one-shot design manner.

Results

The sequential run resulted in measures shown in Figure 2.27. Measures after additional tuning are shown in Table 2.7. Relative improvement compared to the coarse surrogate model is shown in Figure 2.28.
Employing the sequential design for surrogate modeling resulted in a mean relative error of $\epsilon = 0.031$ with standard deviation $\sigma = 0.033$ (considering all outputs). Each output, apart from the **towerMy**, reached the targeted 5% accuracy (considering MRE). Average relative improvement compared to one-shot coarse surrogate model is around 19%. Though these results may not be extraordinary, note only 1051 samples were used to reach them. Note further, that such a model may be build in a couple of hours which makes excellent from price-to-value point of view.

1000 samples fin					Mean relative error
Heterogenetic Sequential number of training samples: 1051 number of validation samples: 321 Individual Measures					
Average model error (considering all measures):	6%				
Average model error (considering only AEE & RRSE):	7%				
	targets reached: 86%				-26%
POWER : Cross Validation - Average Euclidean Error off	0,01	0,05	-79%		-56%
POWER : Cross Validation - Mean Relative Error	0,03	0,05	-35%	1	
POWER : Validation Set - Root Relative Square Error	0,02	0,10	-76%	1	
bladeMx : Cross Validation - Average Euclidean Error off	0,04	0,05	-19%		-30%
bladeMx : Cross Validation - Mean Relative Error	0,03	0,05	-35%	1	
bladeMx : Validation Set - Root Relative Square Error	0,11	0,15	-25%	1	
bladeMy : Cross Validation - Average Euclidean Error off	0,01	0,05	-85%		-83%
bladeMy : Cross Validation - Mean Relative Error	0,01	0,05	-86%	1	
bladeMy : Validation Set - Root Relative Square Error	0,10	0,50	-81%	1	
shaftMx : Cross Validation - Average Euclidean Error off	0,02	0,05	-62%		-70%
shaftMx : Cross Validation - Mean Relative Error	0,01	0,05	-72%	1	
shaftMx : Validation Set - Root Relative Square Error	0,06	0,20	-68%	1	
shaftMz : Cross Validation - Average Euclidean Error off	0,05	0,05	-1%		-22%
shaftMz : Cross Validation - Mean Relative Error	0,04	0,05	-26%	1	
shaftMz : Validation Set - Root Relative Square Error	0,08	0,10	-17%	1	
towerMx : Cross Validation - Average Euclidean Error off	0,05	0,05	8%		-34%
towerMx : Cross Validation - Mean Relative Error	0,03	0,05	-45%	1	
towerMx : Validation Set - Root Relative Square Error	0,16	0,20	-22%	1	
towerMy : Cross Validation - Average Euclidean Error off	0,32	0,05	543%		111%
towerMy : Cross Validation - Mean Relative Error	0,10	0,05	96%	0	
towerMy : Validation Set - Root Relative Square Error	0,23	0,10	126%	0	

Figure 2.27: *Statistics for building the sequential model*

Table 2.7: *Measures of "one-shot coarse surrogate model". Equivalent loads are given in [kNm] and power output in [kW].*

Measure	bladeMx	bladeMy	towerMx	towerMy	shaftMx	shaftMz	Power
maximal sample value	11956.33	8754.42	26168.03	19391.31	8388.66	1306.18	5000.24
maximal surrogate value	11456.44	8675.51	26907.28	19756.81	8306.94	1323.82	5036.57
minimal sample value	2295.66	5806.85	8727.22	2192.74	2826.65	74.57	87.55
minimal surrogate value	2107.54	5887.71	8759.66	2177.15	2780.99	64.43	95.62
Average Eucledian Error	196.70	40.55	321.26	699.50	40.77	18.38	19.10
Root Relative Square Error	0.11	0.09	0.12	0.24	0.05	0.09	0.01
Maximmal Absolute Error	935.56	205.80	2355.36	4423.80	222.98	170.60	131.60
Mean Absolute Error	196.70	40.55	321.26	699.50	40.77	18.38	19.10
Mean Relative Error	0.03	0.01	0.02	0.11	0.01	0.04	0.01
Maximal Relative Error	0.09	0.03	0.14	0.45	0.05	0.38	0.15

Relative improvement

	bladeMx	bladeMy	towerMx	towerMy	shaftMx	shaftMz	Power	AVERAGE
Average Eucledian Error	0,008	0,058	0,099	-0,108	0,391	0,223	0,486	17%
Root Relative Square Error	0,036	0,054	0,096	-0,148	0,467	0,306	0,551	19%
Maximmal Absolute Error	-0,064	0,081	-0,206	-0,760	0,643	0,091	0,465	4%
Mean Absolute Error	0,008	0,058	0,099	-0,108	0,391	0,223	0,486	17%
Mean Relative Error	0,075	0,071	0,192	0,047	0,421	0,291	0,693	26%
Maximal Relative Error	0,228	0,183	-0,061	0,226	0,635	0,444	0,680	33%
AVERAGE	5%	8%	4%	-14%	49%	26%	56%	19%

Figure 2.28: *Relative improvement compared to the coarse surrogate*

2.3.5 Summary and conclusion for surrogate modeling

In the first part, the theory for surrogate modeling has been presented in order to introduce the reader to the topic. Following the introduction, the SUMO toolbox is chosen as the modeling tool and several test have been performed to show its performance and sensitivity to settings.
Based on these studies, an environment has been created in order to set up the simulation platform and establish connection between the surrogate modeling toolbox, and the simulator use for this purpose was HAWC2. The simulations in HAWC2 use the DWM model, so due to their computational requirements, as well as to speed up the computations by running them in parallel, the running environment required the use of a cluster.
The modeling approach is done in a sequential fashion by iteratively building surrogate models and subsequently correcting them by adding new data points. The sampling locations are determined by a smart algorithm, which bases its choice on the (non-)linearity of the surrogate, domain coverage and error w.r.t. the true system response.
The type of a surrogate model is either pre-defined from the available library or found through the use of an evolutionary algorithm. The modeling process consists of first finding the appropriate modeling equation. This is done as a superposition of base functions. The second phase includes the hyper-parameter optimisation, which is most often solved through LSQ fitting.
The target, apart from setting up the framework, was to create a surrogate with 5% mean relative accuracy using up to 1100 points.
Modeling of 7 distinct outputs is performed in 3 stages. First, by creating the coarse model used to test the platform and learn about the behaviour of each output. Second, by running a sequential design and creating a more optimal model in terms of cost-to-value. Finally, by additionally tuning the model by optimising the hyper-parameters, again in one-shot design manner.
In conclusion, surrogate modeling as an idea poses a good substitution for the wake model. It has the advantage of dynamic wake models by providing both load and power outputs. On the other hand, its cost is low compared to full models. The expense comes as reduced accuracy of the model, special preparation requirements and higher initial investment in terms of computational cost.
The created surrogate model is still in its infancy and should be improved once its application is better examined and further tested.

As a continuation of work in the area of surrogate modeling for the purpose of WFLO following steps are proposed:

1. Asses the sensitivity of the objective function to the accuracy of the model and compare the results of current surrogate in WFLO application to the existing look-up tables used as a part of the TOPFARM [22].
   Analysing the sensitivity of the objective function, one should be able to better determine the required precision of each surrogate output. The performance of the surrogate in application to WFLO may be evaluated compared to other surrogate, or not-surrogate wake models. Finally, the optimisation platform encompassing surrogates may be compared to alternatives.
2. Asses the correlation between input variables
   This is done with the aim of analysing if it would be more optimal to separate the modeling of each output or maybe to split it in groups of outputs where the correlation exists.
3. Create a dedicated platform for surrogate modeling of the DWM model
   Having a dedicated tool whose settings may be defined with greater freedom and which may be manipulated according to specific needs must produce better results in terms of modeling. For example, the speed of modeling may be increased by reducing the choice to pre-defined basis functions. Furthermore, the one-shot design approach may be adjusted to tune the existing model, as opposed to building the model from scratch. For this purpose various methods of correcting the local behaviour of the model, such as space mapping, may be implemented. Next, measures may be defined with weights, so that while some average error measurement dominates the modeling and the global fit, some measure of maximal error is used to correct local deviations.

Chapter 3

The Cabling Problem

Cost related to internal grid are estimated to account for 5% of the total investment cost for developing a wind farm [2]. In large wind farms especially, but also in smaller ones, an important task is to find the optimal layout of the internal grid between the turbines. Usually, this is done by finding the cable of minimal length which connects all turbines. Several definitions are found to describe this problem, and due to their similarities the cabling problem can easily be defined using the wrong classification. To eliminate this issue, some typical problem and approaches are described first and their fine differences clearly indicted. In the later part of the section, the cabling problem is precisely defined, and two algorithms for solving it are presented and tested.
Typical problems and approaches usually associated with the cabling problem are:

- **Shortest path**
  The shortest path problem deals with finding the shortest route to a destination using existing "road" infrastructure. The connection between two points thus cannot be arbitrary, but needs to follow existing routes. This problem is equivalent to the use of GPS or tool such as Google Maps for finding directions conditioned by shortest mileage.
  In terms of cabling the wind farm, this approach would be used if the area designated for the farm was "populated" with some kind of obstacles or exclusion zones. Though this is often the case (not for large wind farms), obstacles are not numerous and are more likely to be solved manually. Moreover, the defining characteristic of the shortest path problem is that it has one start and one end.
  As people relevant to this problem, Edsger Dijkstra should be mentioned on the account of many contributions to the field of optimising distances and finding routes.

Dijkstra's algorithm is a specific example of his work and focuses on finding the shortest path. More detailed explanations on Dijkstra's algorithm could be find in publications such as [4] or [42].

- **The travelling salesman**
  The travelling salesman problem (TSP) is one of the (in)famous problems of combinatorial optimisation. It is also sometimes associated with finding the optimal cabling layout between turbines. However, the complexity of TSP is much higher and defining the cabling problem as TSP will end in an over-complicated situation. TSP deals with finding the optimal circular route (shortest time, minimal distance, minimal fuel consumption, ect.) between points. The objective of the travelling salesman is to visit n cities and return to the starting one. The salesman is allowed to visit cities multiple times or use same roads more than once as long as it meets the optimal solution. However, it is proven that the optimal way of travelling to points and back will never use the same roads and will avoid revisiting points. Defined as such there is a total of $(n-1)!$ different possibilities of connecting n points. Half of which are duplicates due to symmetry.
  TSP is defined as NP-hard, and no effective solution exists to find the solution representing the global optimum. Strategies used for solving TSP can be Brute-Force method, Dynamic Programming, Simulated Annealing or Genetic Optimisation for example.

- **Maximum flow**
  The maximum flow problem deals with estimation of a network or system of routes which will ensure the flow of fluid, goods, people ect. to be least congested. This kind of estimates are done for evacuation plans or traffic control. Depending on the precise task, the existing routes are usually used, but the algorithm could also be implemented to plan a new network. This problem is not very well suited to find the cabling layout within the farm.

- **Simulated Annealing**
  Simulated Annealing (SA) is a heuristic approach of solving problems of combinatorial or discrete nature by mimicking the intermittent process of heating and cooling some metal. Similar to the process of genetic algorithms, SA arbitrarily changes solution approximations and either accepts or rejects the change based on the improvement in the objective. Difference compared to some Hill Climbing strategies is that not all adjustments resulting in reduction of the objective are rejected. (In order to climb

the highest hill, one has to descent hills along the way)

- **Minimal Enclosing Circle**

  Another parameter which might be considered when dealing with finding the optimal cabling layout is the point within the farm whose sum of distances to all points is minimal. This problem is termed as the Minimal Enclosing Circle (MEC), since the point represents the centre of the circle with the smallest radius which encloses all points.

  Solution to this problem is a data fitting problem and can be solved very simply by using linear or quadratic programming. Solutions to this problem may be those found by l_1 estimation where the absolute distance to every point is minimised, l_2 estimation where the square distance to each point is minimised, or l_∞ estimation where the maximal distance is minimised.

  Application of finding the minimal enclosing circle may be in determining the position of a substation or an $O\&M$ platform. Example of MEC using l_2 (LSQ) estimation is given for the 30 point layout in Figure 3.1.

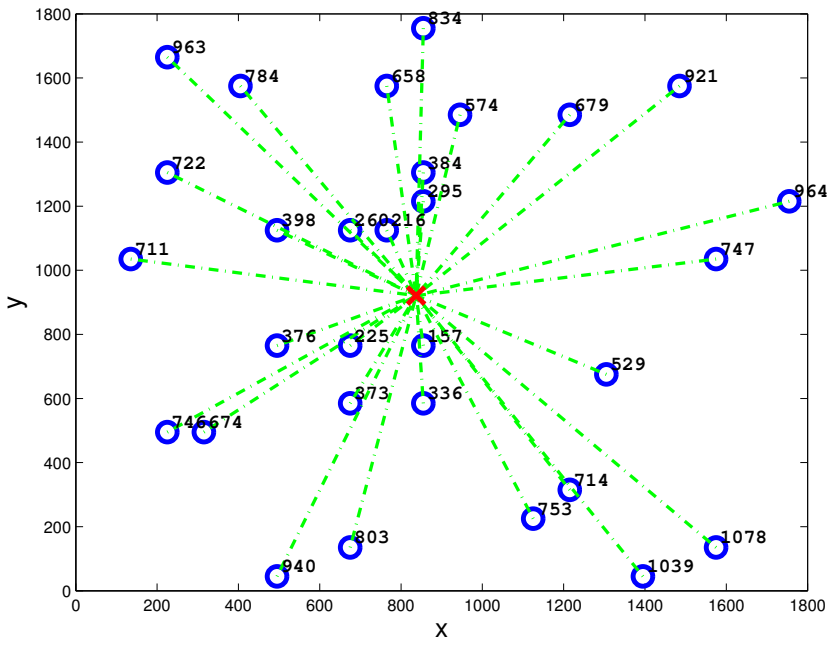


Figure 3.1: *Layout of 30 points (turbines) with the centre of minimal enclosing circle marked as red* $\times$

3.1 The Minimum Spanning Tree and the greedy approach

The cabling of the internal wind farm grid reduces to a problem known as the Minimum Spanning Tree (MST). Such a problem is defined simply as a task of finding the minimal length of total lines connecting arbitrary points. Distinctions compared to the shortest path or the TSP are the following. Shortest path uses existing infrastructure to connect starting point to the target point (1 point to 1 point). Travelling salesman has a task of visiting points from 1 starting point and return to the same. The MST has multiple starting points and its task is to connect, i.e. to build bridges between all points.
Solutions to this problem are given using the greedy approach. Similar to the hill climbing technique, greedy algorithms requires each step to be the best w.r.t. the conditions set by the particular algorithm. Before proceeding to test examples, it is useful to know the optimal cable layout will never contain lines (connections) that are crossing each other.

3.1.1 Prim's algorithm

This algorithm was first developed by Chech mathematician Vojtěch Jarník in 1930 [14], then later in 1957 independently by Robert C. Prim. Furthermore, Edsger Dijkstra should also be be mentioned on account of his contributions to this algorithm. Consequently, Prim's algorithm is also known as DJP algorithm, Jarnik algorithm or the Prim-Jarnik algorithm.
Prim's algorithm solves the problem of "minimum spanning tree" by connecting the starting point (arbitrarily chosen) with the point least distanced from it. The connected point then becomes the starting point, and the search for the shortest distance is continued among the remaining (unconnected) points. From the explanation, it can be noticed that this algorithm operates by connecting consecutive points in "one-after-the-other" manner. If we were to do a task such as this manually by hand, we would do it by connecting the points without raising the hand from the paper, i.e. each point, apart form the first and the last one, will be connected to maximum two other points, one proceeding, other following.
This problem can be solved as rearranging or permuting the set of points. Thus, there can be $(n-1)!$ possibilities of connecting all points (n denotes the number of points). However, this number includes duplicates (symmetrical arrangements) and it can be reduced by n (every starting point can be an ending point). Furthermore, Prim's algorithm chooses the next connecting point by the smallest distance criteria, and the line is continuous. This means that every point has a maximum of two points which the algorithm may use. One of those points will be the point from which the line is coming from, and the other the

one where it will continue. This brings us to the conclusion that once we choose a starting point, the algorithm can move to only one possible location. In the case of n points, starting point will be chosen n times and there will be one duplicate, thus there are only $(n-1)$ possibilities. Moreover, implementing the comparison of the longest section in the resulting cable and the distance between free-ends, additional solutions might drop out.
An example of Prim's algorithm is given in the continuation.
Let's say we have 7 points as shown in Figure 3.2. This means we will have 6 possible layouts.

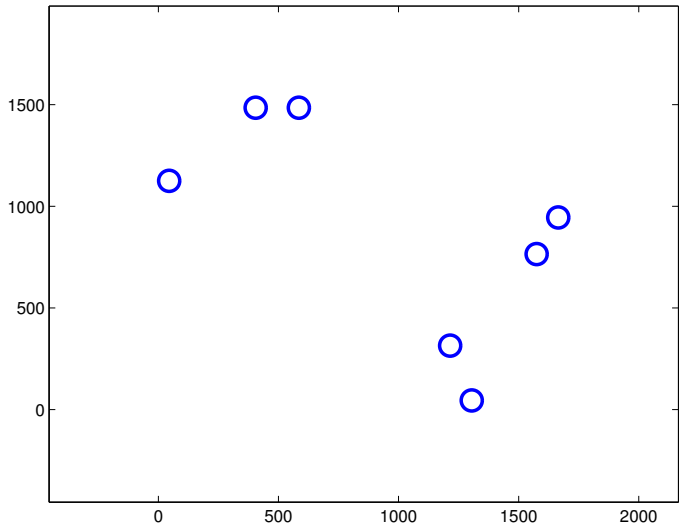
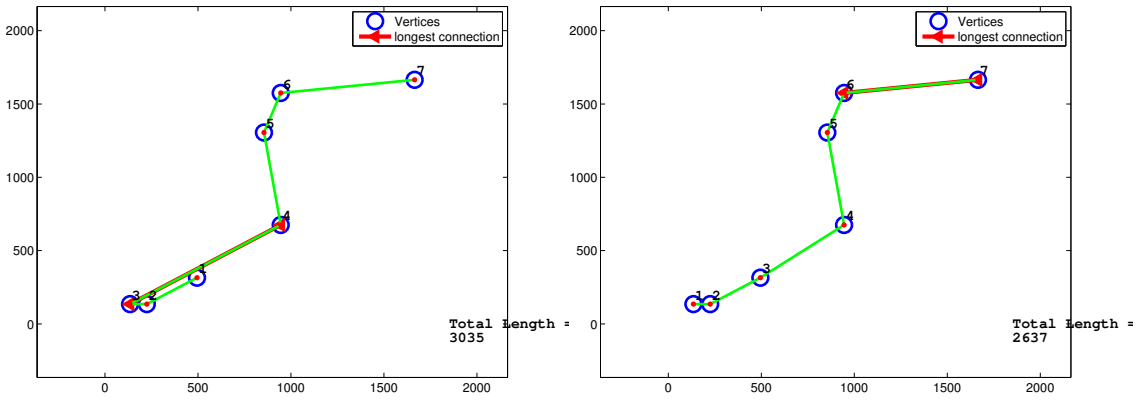


Figure 3.2: *Layout of 7 points (turbines)*

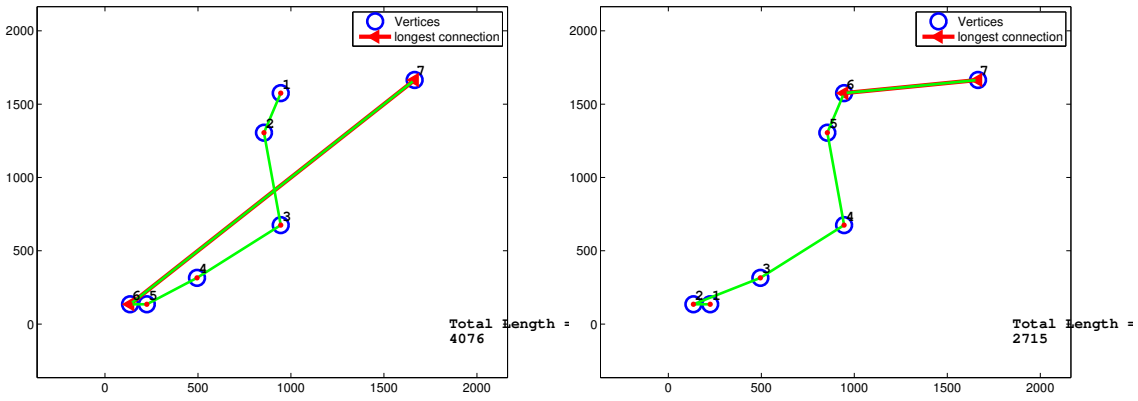
We randomly choose a starting point and run Prim's algorithm from it. The result is shown in Figure 3.3a. The longest distance between points in Figure 3.3 is highlighted by a red line. This serves the purpose of comparing the longest connection between points with the distance between "free-ends". In case the length between the starting and ending point is shorter than the one found in the cable, algorithm will swap those two connections (this option is switched off in test example). This issue can be seen by examining Figure 3.3c. Notice first the lines are intersecting each other, thus we know this cannot be an optimal layout. Secondly, notice the longest distance $6-7$ (highlighted in red) is clearly longer than that which would connect points labelled $1-7$. If the swap was to be made, resulting layout would be identical to that seen in Figure 3.3b.
Next, notice the layouts in Figure 3.3b and Figure 3.3f are identical, although their starting points are different. This is the example of a symmetric layout. We have thus reduced 7 layouts to 5 different ones. Furthermore, connection $4-5$ in Figure 3.3e could be swapped with $1-7$ to make the total cable length bit shorter. This would be identical to layout

seen in Figure 3.3g. Thus we have reduced this example to 4 possible layouts. The optimal one using Prim's algorithm is found in Figure 3.3f or b. Summary of connections is given in Table 3.1.



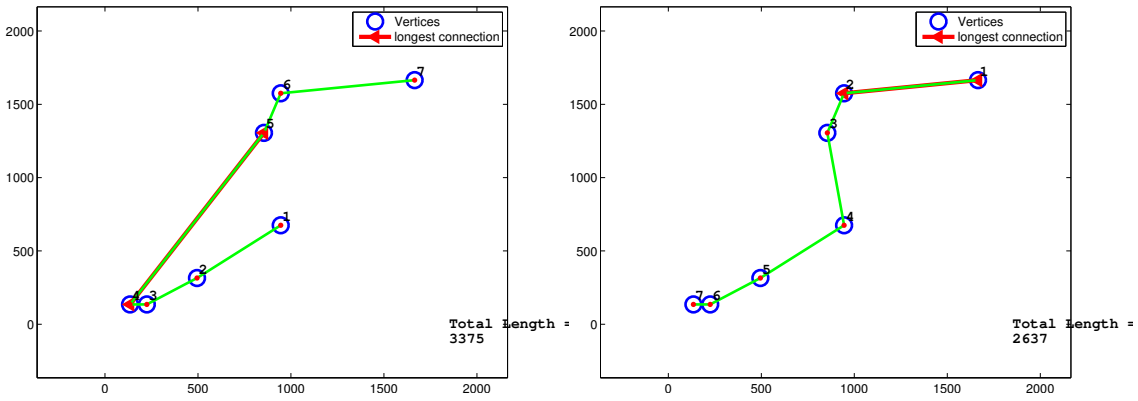
(a) 1st run, cable length 3035

(b) 2nd run, cable length 2637



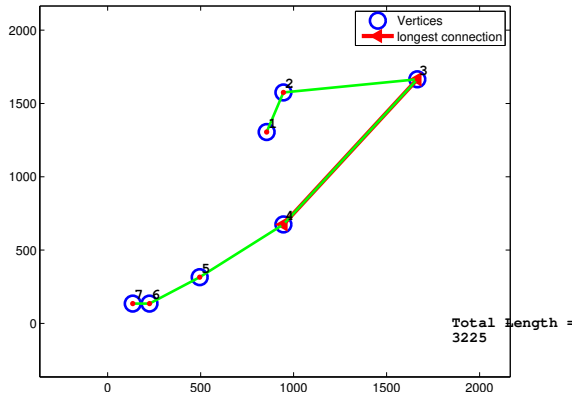
(c) 3rd run, cable length 4076

(d) 4th run, cable length 2715



(e) 5th run, cable length 3375

(f) 6th run, cable length 2637



(g) 7th run, cable length 3225

Figure 3.3: *Prim's algorithm ran on points shown in Figure 3.2*

Table 3.1: *Summary for cabling layout shown in Figure 3.3e*

connecting	(x1, y1)	(x2, y2)	length
1 2	(135, 135)	(225, 135)	90
2 3	(225, 135)	(495, 315)	324.5
3 4	(495, 315)	(945, 675)	576.28
4 5	(945, 675)	(855, 1305)	636.4
5 6	(855, 1305)	(945, 1575)	284.6
6 7	(945, 1575)	(1665, 1665)	725.6

Could we determine the optimal starting point without testing all possibilities? This may be possible on a small scale problem, however, let's take the example of having 30 points as in Figure 3.4 or 100 points Figure 3.5. Situation then gets messy.

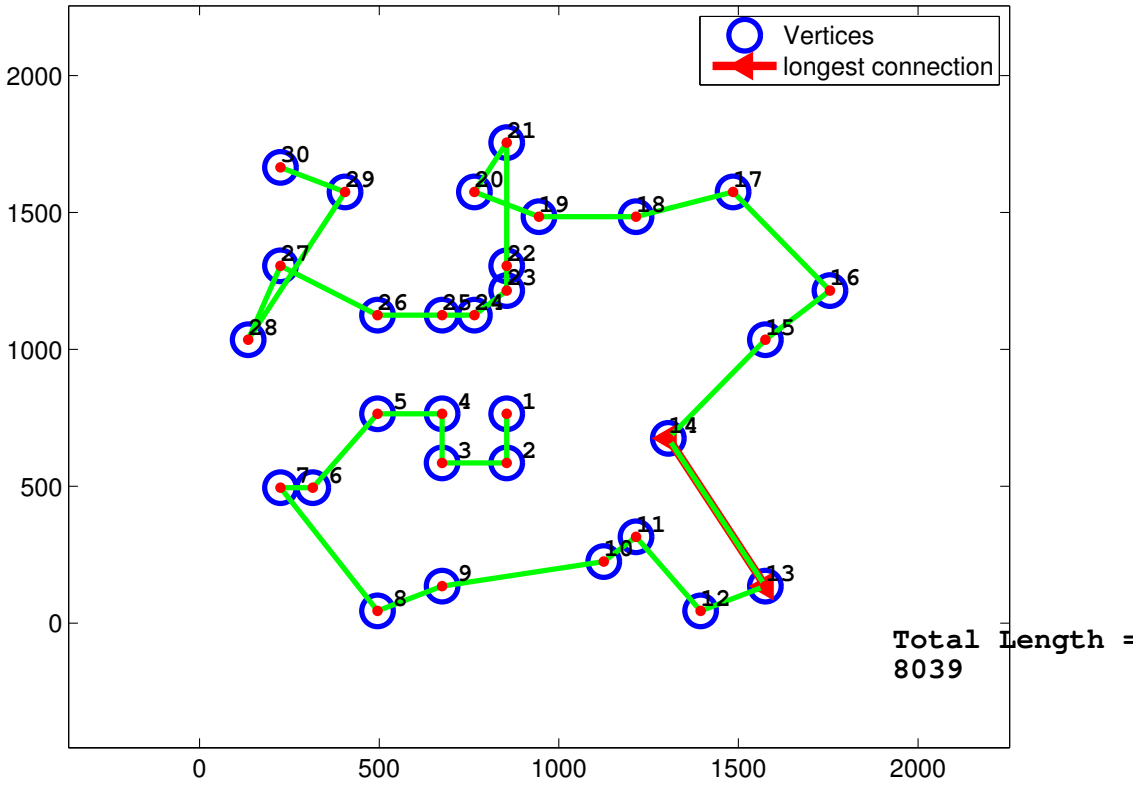


Figure 3.4: *Layout consisting of 30 points, connected using Prim's algorithm*

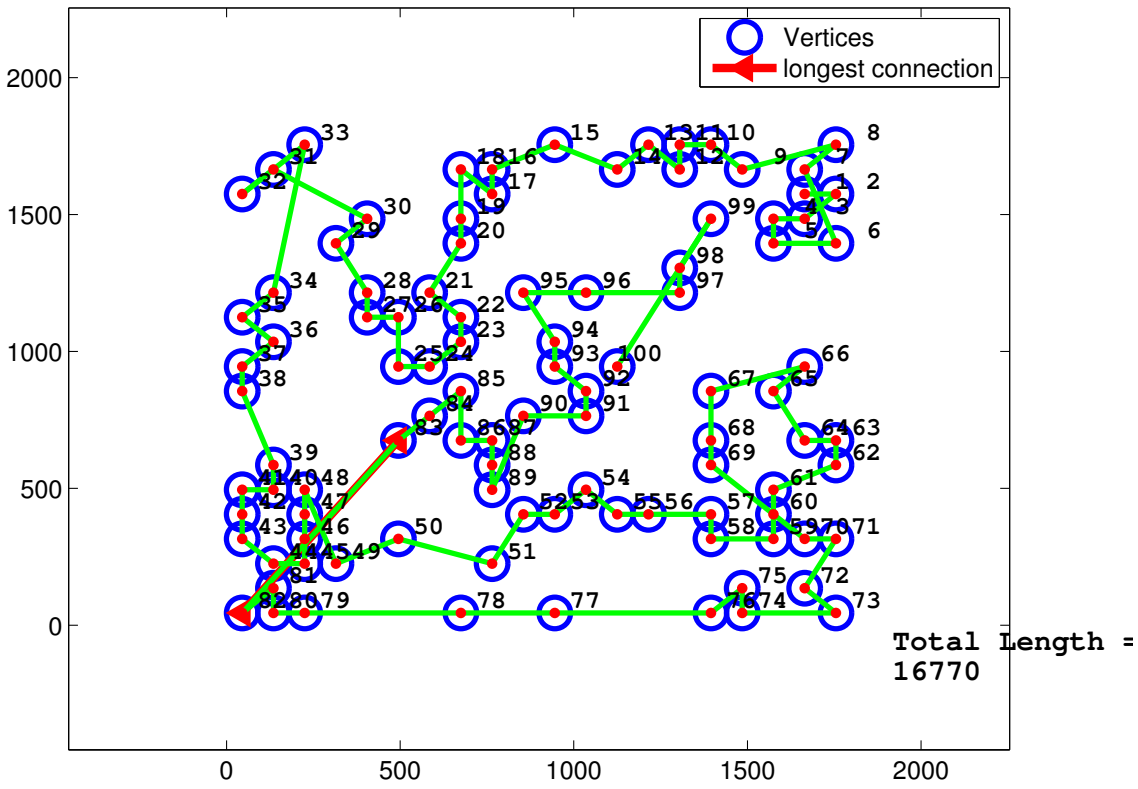


Figure 3.5: *Layout consisting of 100 points, connected using Prim's algorithm*

The problem with Prim's algorithm is it does not allow a "hand to be lifted from the papir". Knowing we are dealing with the problem called the minimum spanning tree, one can easily come to the conclusion (by looking at any nearby tree and noticing many branches) that this problem can be solved in a better way, by allowing the connections to branch from each point. Prim's algorithm in terms of greed can be imagined as a situation where we have to grab as many candies from the table, but using only one hand. Competing, for example, with an octopus, we don't stand much chance.

3.1.2 Kruskal's algorithm

Krusal's algorithm [19] is an improvement of Prim's in a way that it connects two points which are least apart. That way we are free to connect any two points in the layout, no matter which points have been connected previously. Next thing to notice is there will be no difference in the result, regardless on the number of iterations we perform (same points are always connected). The solution of this algorithm is thus unique.
The tricky part in implementing this algorithm lies in ensuring that points are not connected too many times. Take for example the layout shown in Figure 3.6. One should

know that for n points, there are $\frac{n\cdot(n-1)}{2}$ possible connections and in order to connect n points, only $n-1$ are needed.

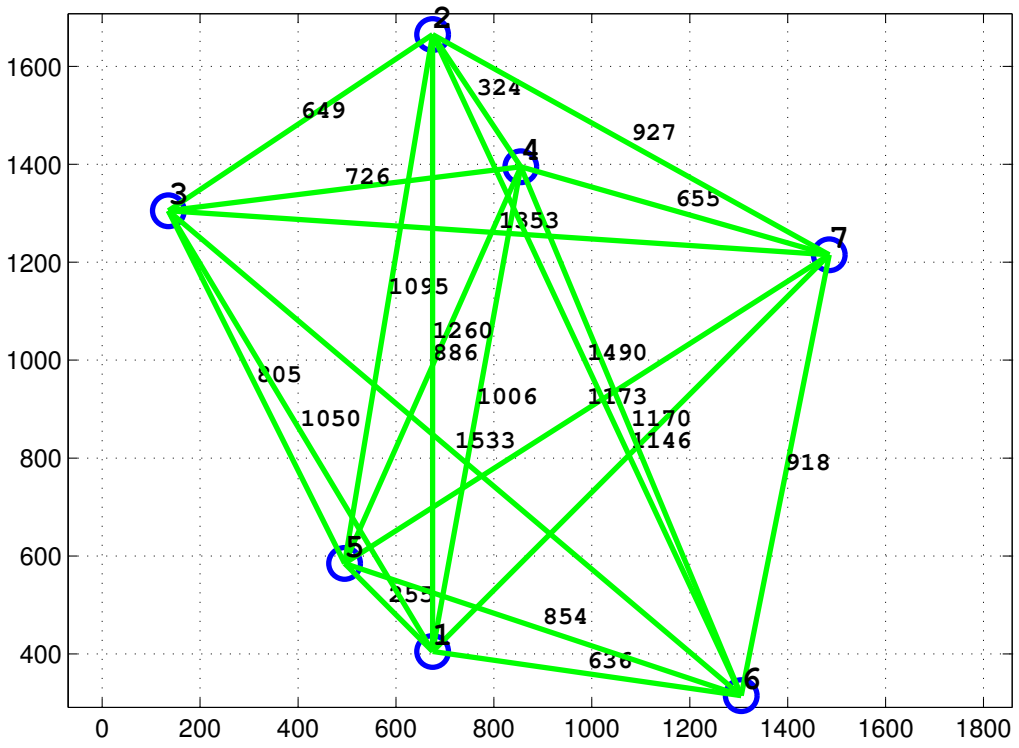
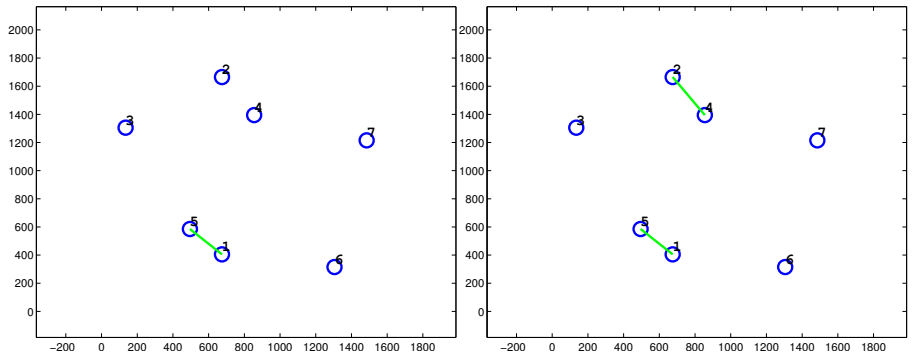


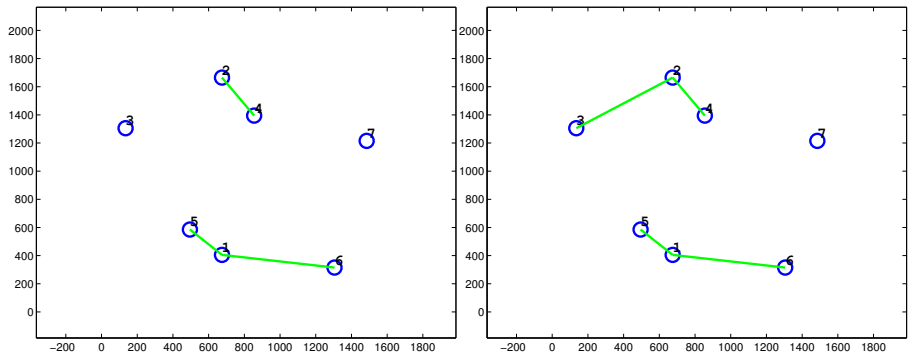
Figure 3.6: *Layout consisting of 7 points, all possible connections.*

Kruskal's algorithm will start by choosing the shortest connections first. The steps for connection a 7 point layout are shown in Figure 3.7. Notice the 6th connection is made between points labelled 3 and 4. This is indeed the shortest remaining connection, but it is unnecessary since it establishes no new members in the union of connected points.



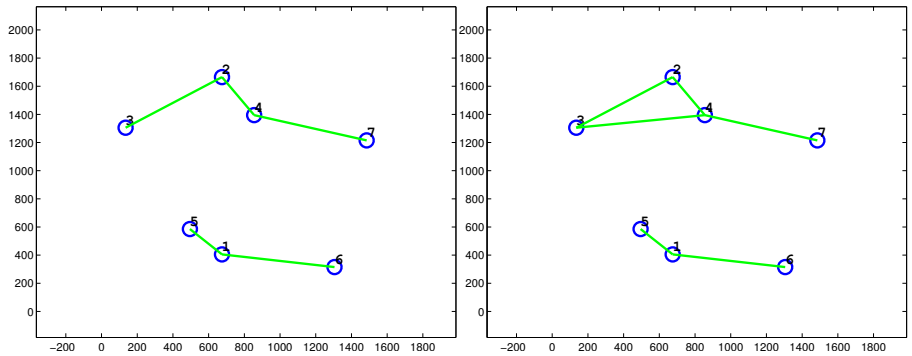
(a) 1st connection

(b) 2nd connection



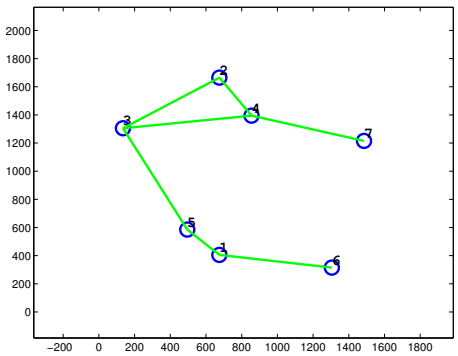
(c) 3rd connection

(d) 4th connection



(e) 5th connection

(f) 6th connection



(g) 7th connection

Figure 3.7: *Steps in Kruskal's algorithm ran on 7 points, effect of over-connecting*

In order to explain how this issues was solved, implementation of Kruskal's algorithm is documented next.

- First, n number of points (x and y coordinate) are put in a $(n - by - 2)$ matrix (termed POINTS matrix) where every point is mapped according to the row in which it is.
- Second, a matrix containing distances of all connections between the points is created (termed DIST matrix). Notice the first point (arbitrarily chosen) will have $n - 1$ possible connections, the second point will have $n - 2$ (connection with the first point is already covered), third $n - 3$, and so on. Last point will have no connections, since all other points are already connected to it. We thus have a $n \times (n - 1)$ matrix, where values in lower triangle are non-existing, i.e. matrix is upper triangular.
- The mapping of the matrix is given as follows. First points in the connection is found in the POINTS matrix at row equal to the column number of the DIST matrix. Second point in the connection is found in the POINTS matrix at row number equalling the sum of its column and row position in DIST matrix. For example, element in DIST matrix in the 4th row and 2nd column is a connection between 2nd and 6th point in the POINTS matrix. This is also shown schematically in Figure 3.8.

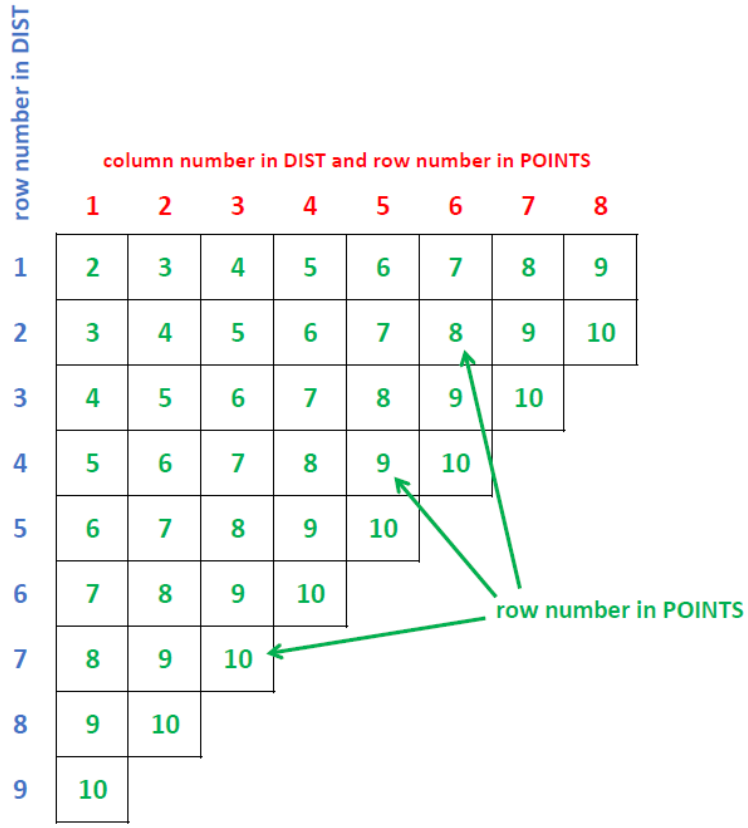


Figure 3.8: *Mapping of distances between points.*

- For each of the points a vector is created containing the the number of the row where the point is found in POINTS matrix. This is the point ID and will serve to mark connections with other points. Every time two points connect, they inherit each other's connections. This means if point 1 is connected to points $5, 8, 9$ and point 12 to $6, 4$ then by connecting points 1 and 12, the vector containing their connections will be $1, 4, 5, 6, 8, 9, 12$ (in numerical order).
  Growth of connections is shown for example in Figure 3.7. Note again, the connection in the 6th step should be avoided.

  1. step
     point1 connections: [1]
     point2 connections: [2]
     point3 connections: [3]
     point4 connections: [4]
     point5 connections: [5]
     point6 connections: [6]
     point7 connections: [7]

  2. step
     point1 connections: [1, 5]
     point2 connections: [2]
     point3 connections: [3]
     point4 connections: [4]

point5 connections: [1, 5]
point6 connections: [6]
point7 connections: [7]

3. Two step
   point1 connections: [1, 5]
   point2 connections: [2, 4]
   point3 connections: [3]
   point4 connections: [2, 4]
   point5 connections: [1, 5]
   point6 connections: [6]
   point7 connections: [7]

4. step
   point1 connections: [1, 5, 6]
   point2 connections: [2, 4]
   point3 connections: [3]
   point4 connections: [2, 4]
   point5 connections: [1, 5, 6]
   point6 connections: [1, 5, 6]
   point7 connections: [7]

5. step
   point1 connections: [1, 5, 6]
   point2 connections: [2, 3, 4]
   point3 connections: [2, 3, 4]
   point4 connections: [2, 3, 4]
   point5 connections: [1, 5, 6]
   point6 connections: [1, 5, 6]
   point7 connections: [7]

6. step
   point1 connections: [1, 5, 6,7]
   point2 connections: [2, 3, 4]
   point3 connections: [2, 3, 4]
   point4 connections: [2, 3, 4]
   point5 connections: [1, 5, 6, 7]
   point6 connections: [1, 5, 6, 7]
   point7 connections: [1, 5, 6, 7]

7. step (no new points!)
   point1 connections: [1, 5, 6,7]
   point2 connections: [2, 3, 4]
   point3 connections: [2, 3, 4]
   point4 connections: [2, 3, 4]
   point5 connections: [1, 5, 6, 7]
   point6 connections: [1, 5, 6, 7]
   point7 connections: [1, 5, 6, 7]

8. step
   point1 connections: [1, 2, 3, 5, 6, 7]
   point2 connections: [1, 2, 3, 5, 6, 7]
   point3 connections: [1, 2, 3, 5, 6, 7]
   point4 connections: [1, 2, 3, 5, 6, 7]
   point5 connections: [1, 2, 3, 5, 6, 7]
   point6 connections: [1, 2, 3, 5, 6, 7]
   point7 connections: [1, 2, 3, 5, 6, 7]

- After each connection vectors containing connections of each point are translated to coordinates for DIST matrix, and that entry is erased so that it cannot be used as a new connection. This serves to eliminate the possibility of bridging points already connected.

Example of fully functioning Kruskal's algorithm, such as the one described above, is given for the 30 points with coordinates as in Figure 3.4. The resulting grid layout is shown in

Figure 3.9. Same was done for the 100 point layout on which Prim's algorithm resulted in Figure 3.5 and Kruskal's in Figure 3.10. Improvement of Kruskal's algorithm compared to Prim's in these examples is 934m and 4030m shorter cable, respectively.

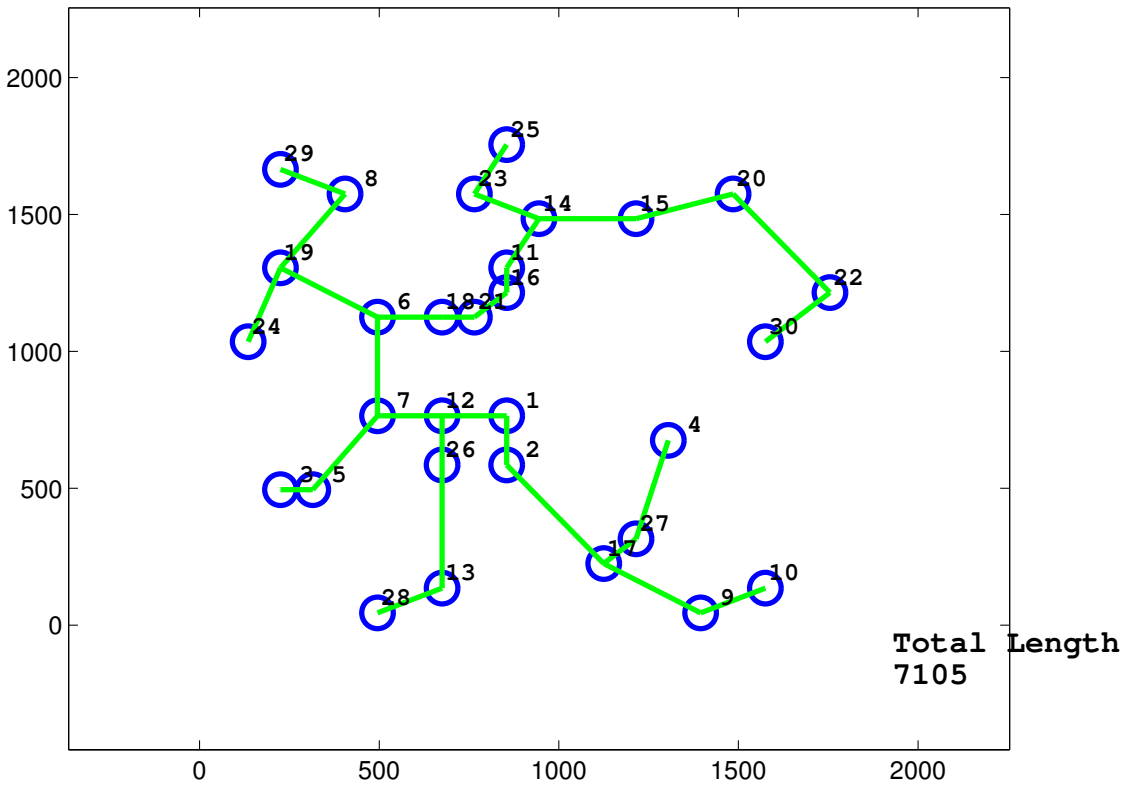


Figure 3.9: *Kruskal's solution for 30 point layout*

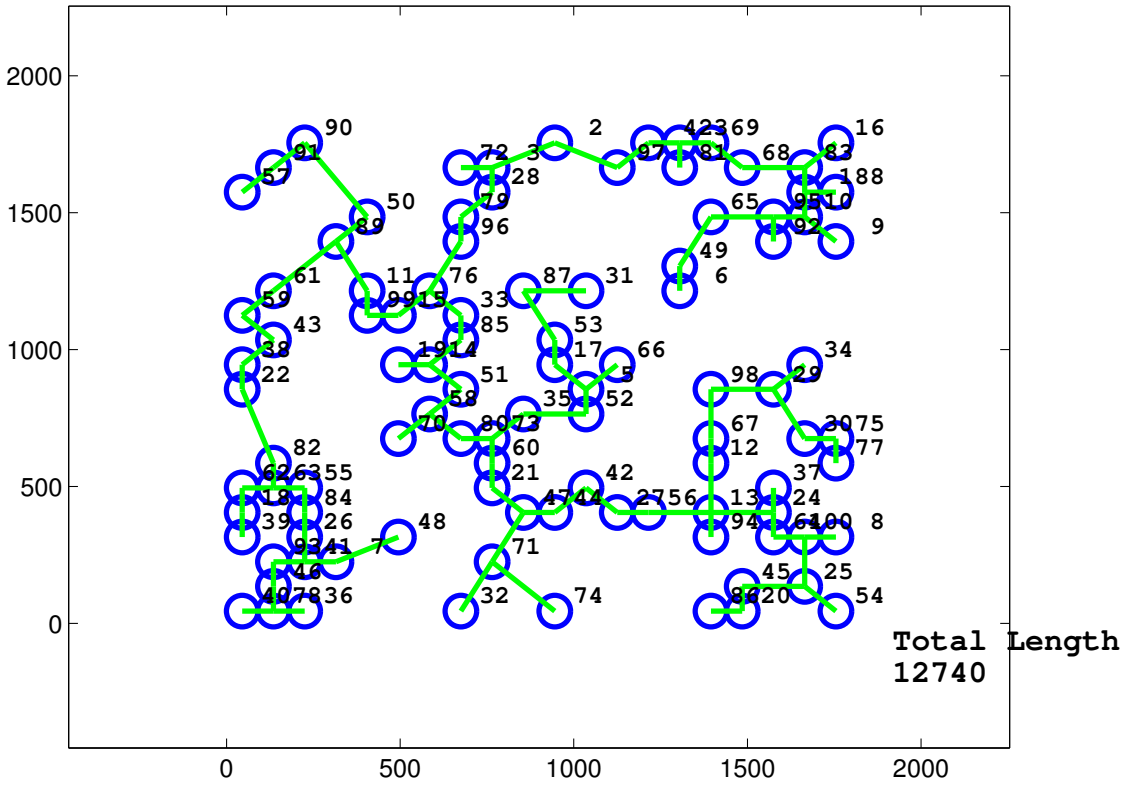


Figure 3.10: *Kruskal's solution for 100 point layout*

3.2 Conclusion and comments

Following conclusion are drawn from the theory and examples presented above.
Prim's algorithm draws a continuous line between points by choosing one starting point and sequentially connecting to the one nearest. Final layout thus depends on the choice of where to start, and it has been shown there is a maximum of $(n-1)$ different solutions. Kruskal's algorithm connects the points least apart with the condition that connection to a new point has to be established in every step. As a consequence, the lines are allowed to branch. Solution resulting from Kruskal's algorithm is unique and does not depend on the starting point. Due to its greedy nature it can be reasoned that this solution represents the global optimum. This is because any other connection not included in the layout has a higher length compared to those found in the solution. This means that total length of the cable can be reduced only if two or more connections in the layout are swapped with lower number of connections not included in the solution. This is however impossible since the minimum number of lines connecting n points is $n-1$, same as drawn in the solution. In case of layouts which encompass zones through which cables cannot be laid, the solution may be found using Graham's scanning algorithm [12].

Chapter 4

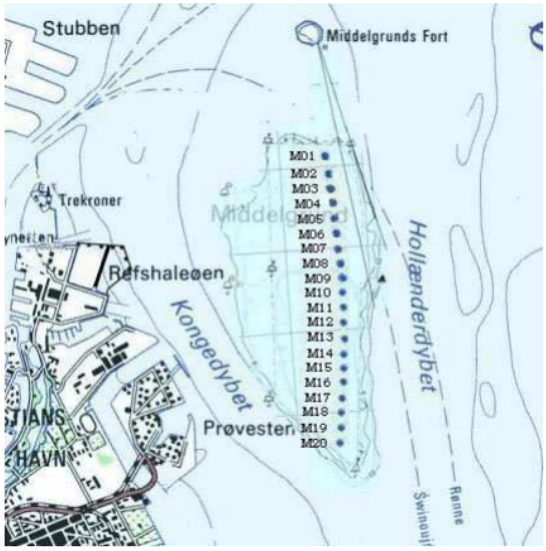
Optimising wind farm layouts

This chapter presents a framework for optimisation of wind farm layouts. The framework consists of a module for wind farm analysis which encompasses the surrogate models created in subsection 2.3.4, and an optimisation platform that is present bit later. To start with, a short example is made by analysing scaled Middelgrunden wind farm. Later on, same wind farm is optimised w.r.t. the NPV.

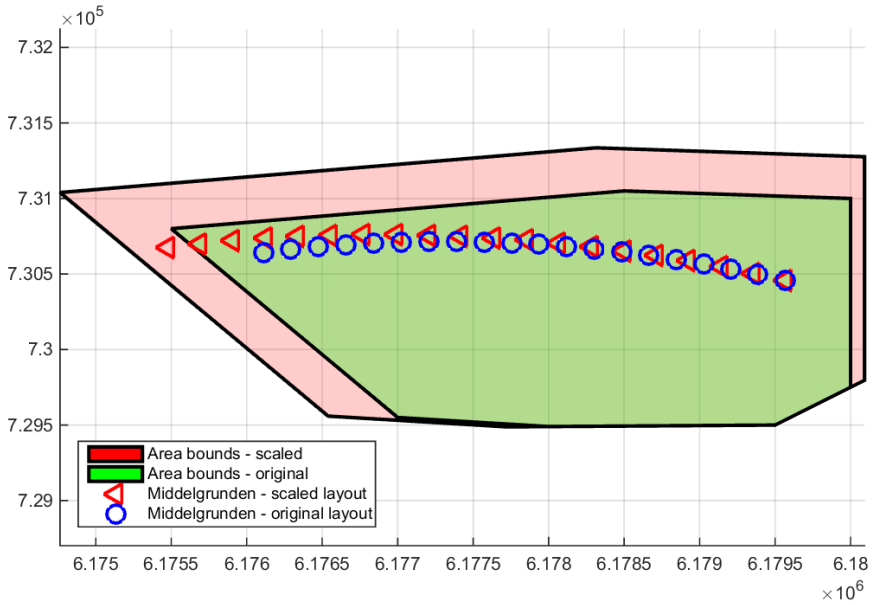
4.1 Wind farm analysis - Middelgrunden

Middelgrunden wind farm consists of 20 Bonus 2.0 MW/76 wind turbines located 2km offshore of Copenhagen harbour as shown in Figure 4.1a,b.
The turbine model available for this project is NREL 5MW/90 wind turbine. Thus, the original Middelgrunden layout is scaled w.r.t. the turbine diameter. The scaled layout is shown in Figure 4.1c. Note for the later use in the layout optimisation, the area boundaries are also scaled.
Presented figures and results are normalised w.r.t. the results of simulating the wind turbine in free wind conditions at 10m/s.

(a) Areal photo



(b) Map



(c) Layout. (North is to the left)

Figure 4.1: *Middelgrunden wind farm*

Middelgrunden site is characterised by the wind distribution as shown in Table 4.1. The wind rose is graphically shown in Figure 4.2.

Table 4.1: *Weibull distributions parameters w.r.t. wind direction*

Sector	1	2	3	4	5	6	7	8	9	10	11	12
Wind direction	0°	30°	60°	90°	120°	150°	180°	210°	240°	270°	300°	330°
Scale parameter	8.5	7.6	7.7	8.2	9	8.4	8.3	7.6	7.7	7.9	7.6	6.7
Shape parameter	2.01	2.32	3.09	2.19	3	2.73	2.21	2.32	2.76	2.72	2.42	2.05
Mean wind speed	7.54	6.77	6.86	7.27	8.02	7.44	7.34	6.74	6.87	7.07	6.76	5.92
Sector probability	0.07	0.04	0.09	0.07	0.15	0.08	0.12	0.08	0.1	0.1	0.06	0.04

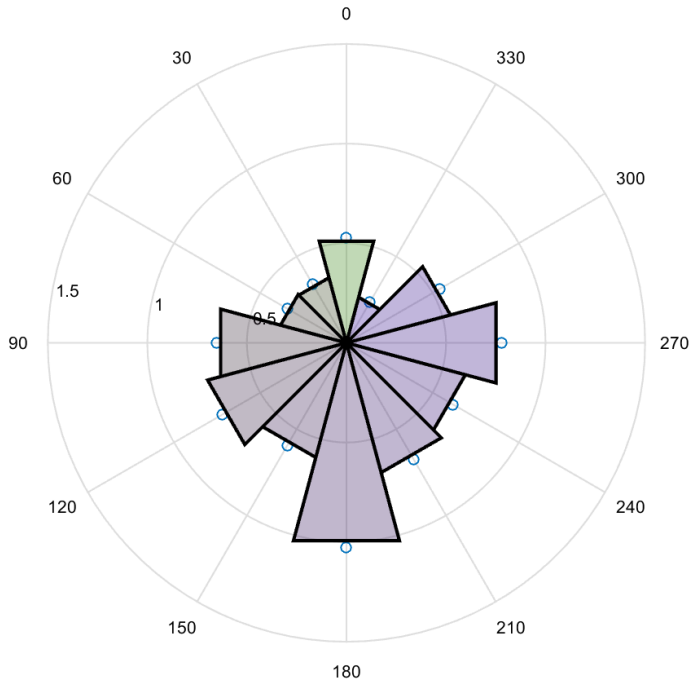
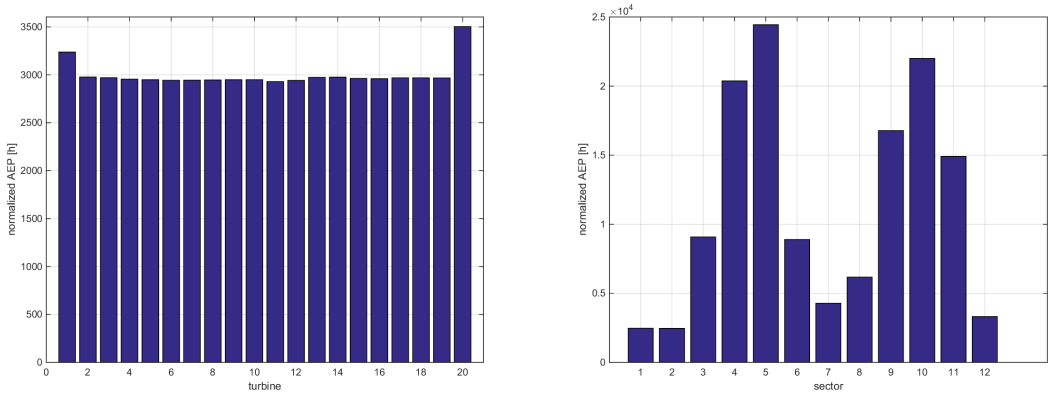


Figure 4.2: *Wind rose for Middelgrunden site. Mean wind speed for the sector is scaled with the probability of the wind direction.*



(a) Turbine-wise (b) Sector-wise

Figure 4.3: *Normalized AEP for Middelgrunden wind farm. 12 50°-wide sectors.*

Wind farm analysis is done by evaluating the performance of each individual wind turbine. This is done for 12 wind directions, where each direction is associated with a sector. The concept of sector is necessary in order to identify which turbines cause the wake effect. Because the current version of DWM model does not take into account effect of merged wakes, the turbine causing the wake is chosen as the one nearest to the turbine being evaluated.

An analysis of individual turbines is shown for three characteristics locations as shown in Figure 4.4. On the first glance, turbine at type "A" location experiences free wind conditions from all, but 180° direction. Turbines at type "B" location experience wake conditions in sectors 1 and 7 (wind from 0° and 180°, respectively) and free wind conditions in the remaining sectors. Turbine at type "C" location experiences free wind conditions from all directions, but wind from 0°. However, this reasoning is valid only for the approach where each sector considers turbines within the angle of ±15°. Due to the effect of meandering and expansion, the wake will also be present from sources originating outside the 30°-wide sector. For example, turbine at type "A" locations will experience partial wake conditions for wind direction of 210° even though Figure 4.4 shows no turbines in that sector. This is the reason why for each of 12 wind directions, sector needs to consider possible sources within at least ±25° (chosen based on the behaviour of the surrogate model). The effect of "widening" the sector and influence to performance of the turbines is shown in Figure 4.5. Notice the 12 30°-wide sector neglects the wake effect for some wind directions.

The code written for the analysis of the wind farm is made in a way that the wake of only the nearest turbine in a sector is considered to have effect. In the case when several turbines appear in the sector, such as seen in Figure 4.4, this assumption will hold for

the power output. However, it may occur the distance is not critical parameter in terms of loading. In case when the turbine causing the wake is near, the meandering effect may not yet be significant since the meandering develops over a certain distance. As a consequence, the sources (turbines) positioned further away might in fact have a more emphasised impact to the loading. Nevertheless, this effect is registered for few limited situations. If we would want to make sure the worst case scenario is considered, each of the possible wake sources should be considered for each output and then the results chosen accordingly. This would, however, mean that depending on the output, the turbine may consider different sources. This was thought to be impractical and consequently was not implemented in the code.

As a consequence of normalizing the power production, the results of AEP are given in units of hours.

Normalized power curves for turbines at the representative locations, and given for each sector, are shown in Figure 4.6. AEP distribution over sectors for the same locations is shown as a "rose" in Figure 4.7. Normalized AEP for each wind speed bin w.r.t. the sector and location is shown in Figure 4.8.

Sector-wise and turbine-wise distribution of energy production for the whole farm is shown in Figure 4.3.

The lifetime equivalent load at different parts of the turbine is shown in Figure 4.9 for location type "A", in Figure 4.10 for location type "B", and in Figure 4.11 for location type "C". The results are scaled w.r.t. the maximum values to indicate the change due to the wake effect.

It is important that the interpreting of the results is done by considering also the probabilities of each wind direction. For this reason, the results overlooking the probability of wind speed and direction are plotted over the real values for turbine at type "B" location. This is seen as outer rose in Figure 4.10.

The wake causes higher loading compared to free wind conditions, the amplitude is, however, different considering different components. For example, highest effect is seen for the equivalent bending moment of the tower bottom in the "front-backwards" direction. Smallest effect are at the seen at the blade root.

Fatigue of components for all turbines is shown in Figure 4.12. These results will be averaged for every component and used as weighted sum to compute the fatigue degradation factor as explained in subsection 1.3.2.

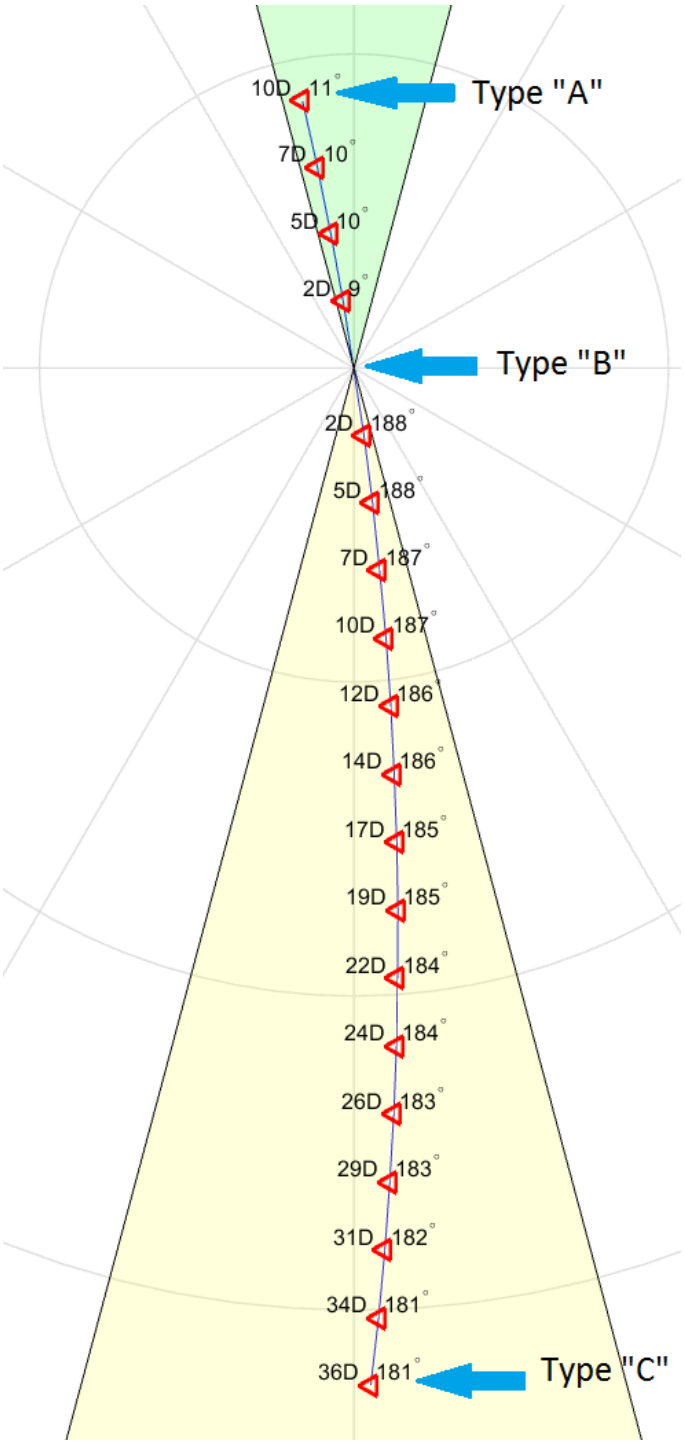


Figure 4.4: *Representative turbine locations. The spacings and angles are given as seen from one of the turbines at type "B" location.*

(a) Type "A" turbine location. 12 30°-wide sectors

(b) Type "B" turbine location. 12 30°-wide sectors

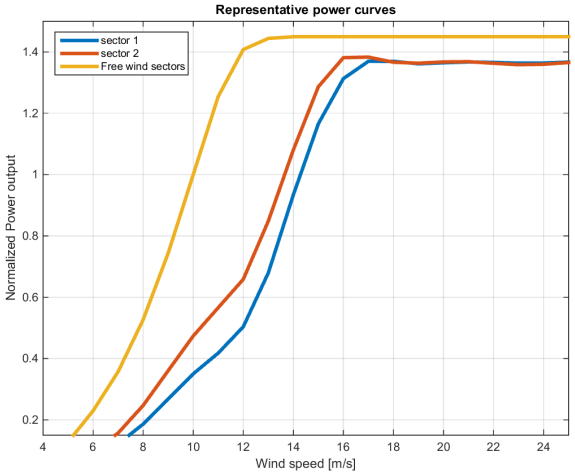
(c) Type "C" turbine location. 12 30°-wide sectors

(d) Type "A" turbine location. 12 50°-wide sectors

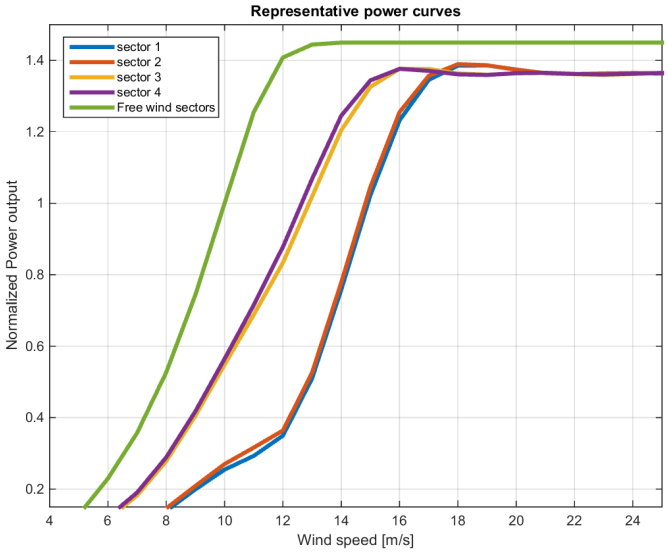
(e) Type "B" turbine location. 12 50°-wide sectors

(f) Type "C" turbine location. 12 50°-wide sectors

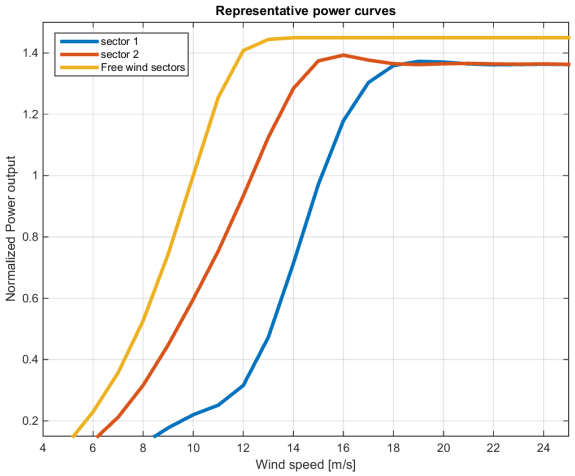
Figure 4.5: *Effect of widening the sector*



(a) Type "A" turbine location

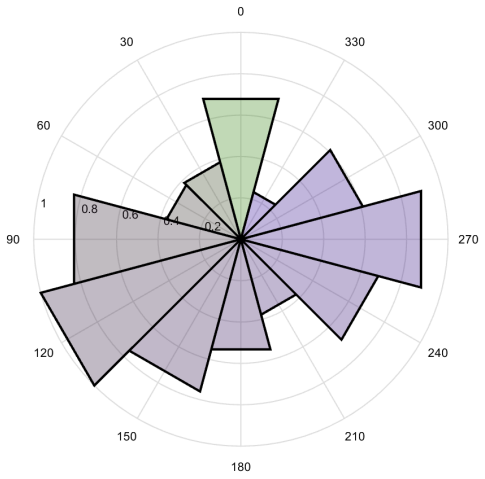


(b) Type "B" turbine location

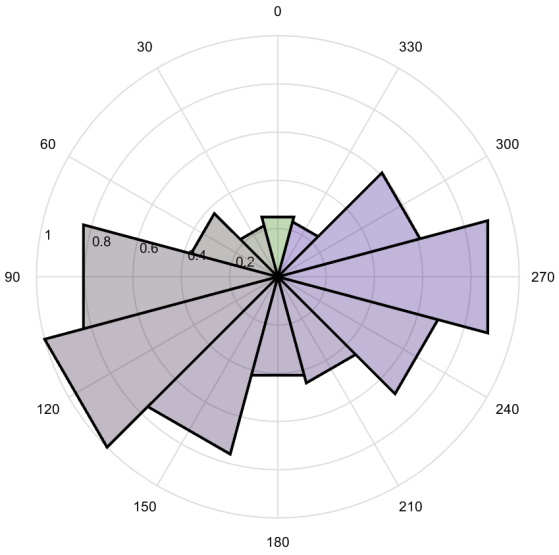


(c) Type "C" turbine location

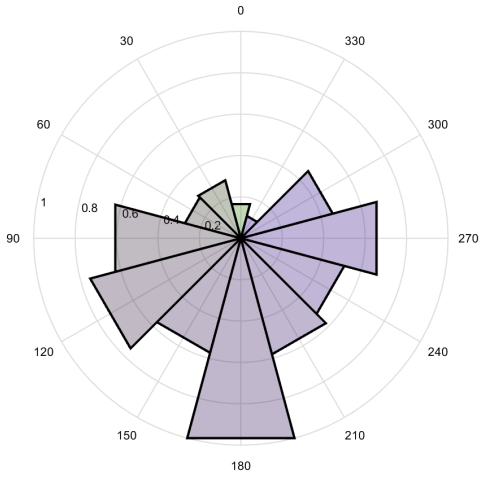
Figure 4.6: *Power curves*



(a) Type "A" turbine location

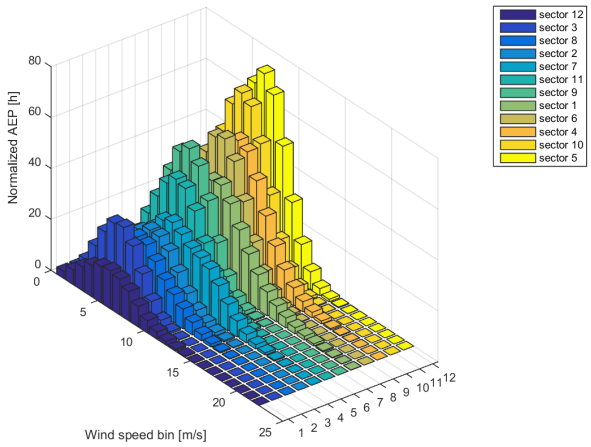


(b) Type "B" turbine location

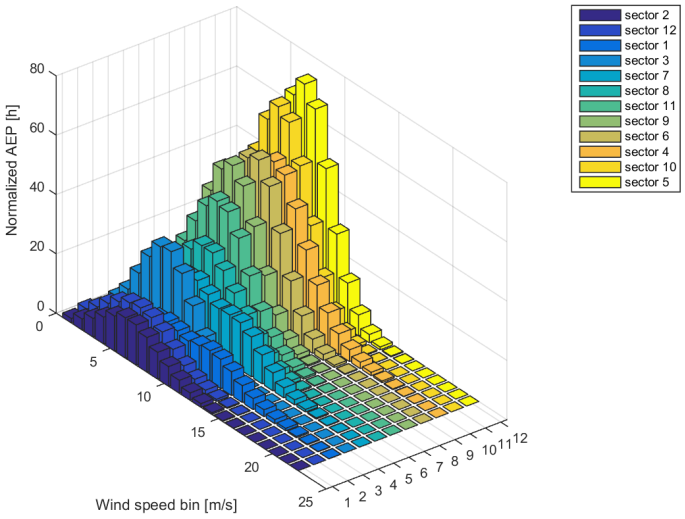


(c) Type "C" turbine location

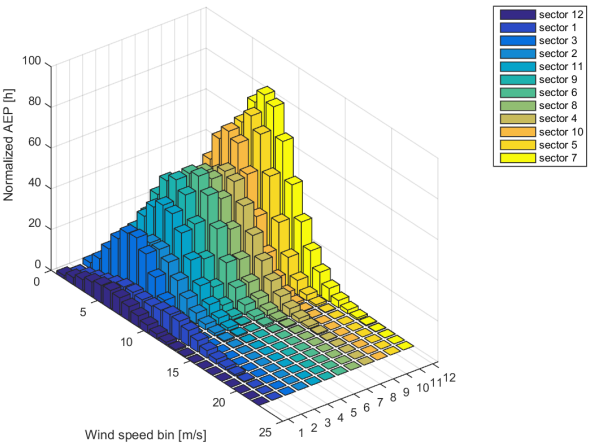
Figure 4.7: *Distribution of AEP over sectors*



(a) Type "A" turbine location

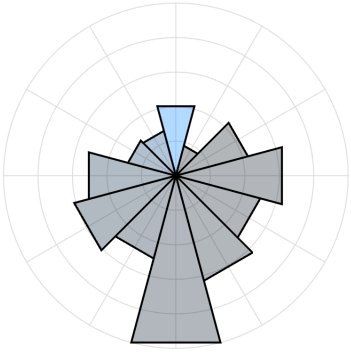


(b) Type "B" turbine location

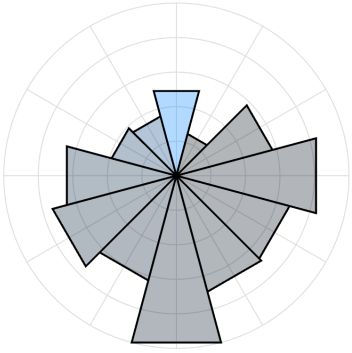


(c) Type "C" turbine location

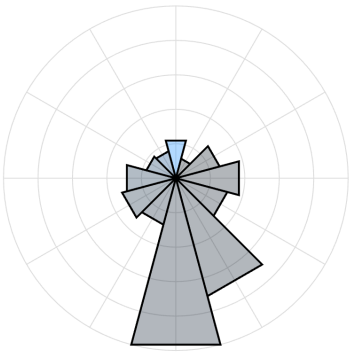
Figure 4.8: *Normalized AEP w.r.t. wind speed bin*



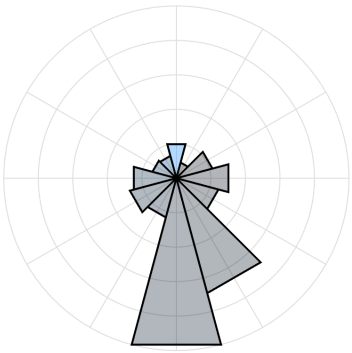
(a) Equivalent load (Mx) at the blade root



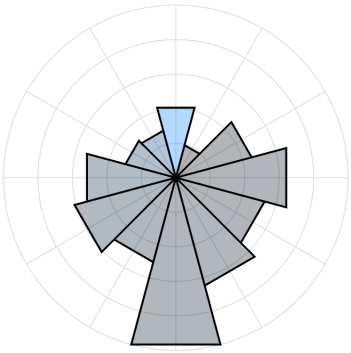
(b) Equivalent load (My) at the blade root



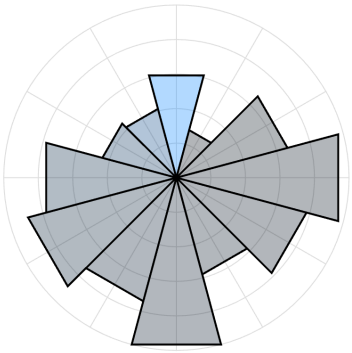
(c) Equivalent load (Mx) at the tower bottom



(d) Equivalent load (My) at the tower bottom

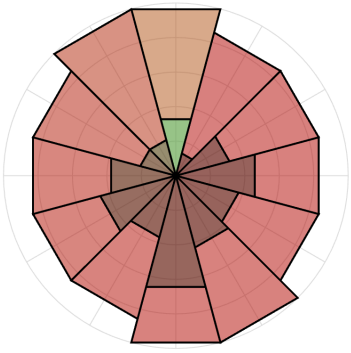


(e) Equivalent load (Mx) at the shaft

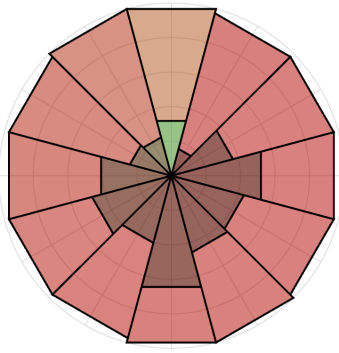


(f) Equivalent load (Mz) at the shaft

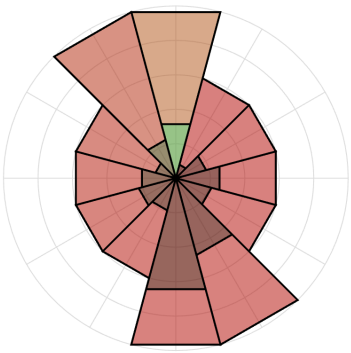
Figure 4.9: *Equivalent load moments for the turbine at type "A" location*



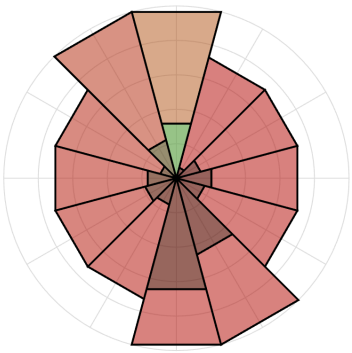
(a) Equivalent load (Mx) at the blade root



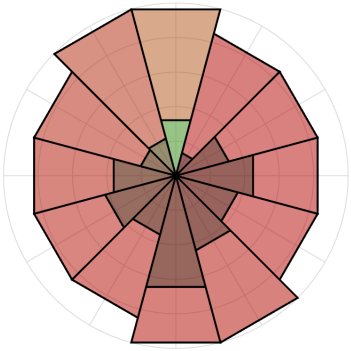
(b) Equivalent load (My) at the blade root



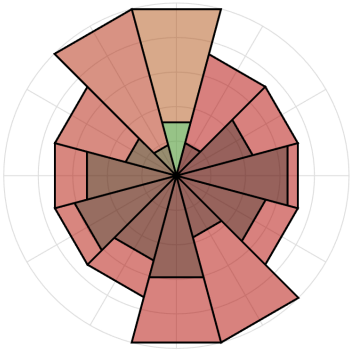
(c) Equivalent load (Mx) at the tower bottom



(d) Equivalent load (My) at the tower bottom

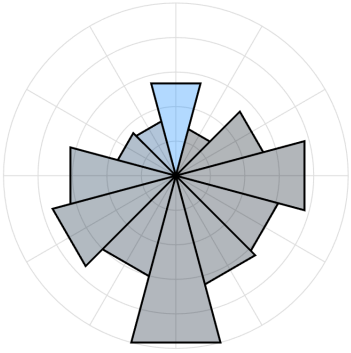


(e) Equivalent load (Mx) at the shaft

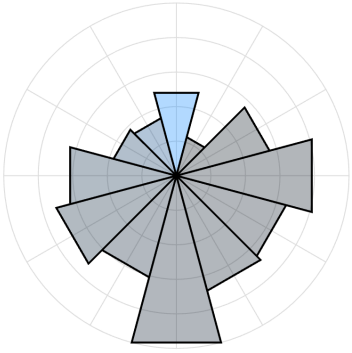


(f) Equivalent load (Mz) at the shaft

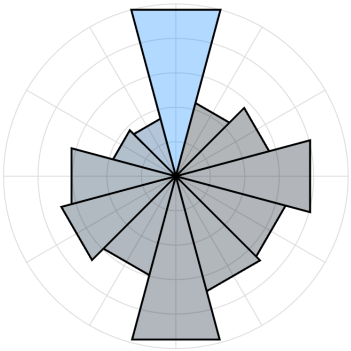
Figure 4.10: *Equivalent load moments for the turbine at type "B" location. The outer distribution is not scaled with probability of each wind direction nor with Weibull distribution fro every sector.*



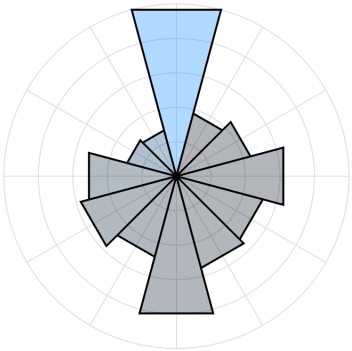
(a) Equivalent load (Mx) at the blade root



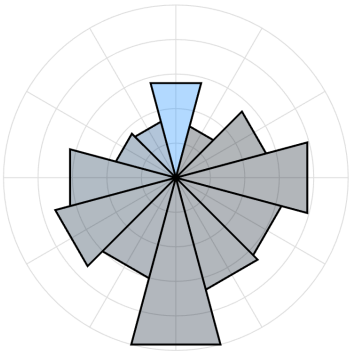
(b) Equivalent load (My) at the blade root



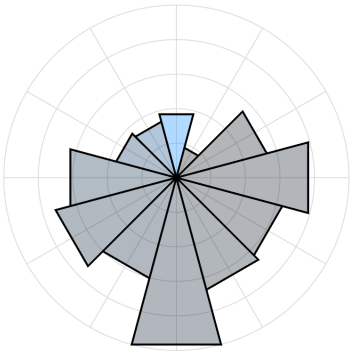
(c) Equivalent load (Mx) at the tower bottom



(d) Equivalent load (My) at the tower bottom



(e) Equivalent load (Mx) at the shaft



(f) Equivalent load (Mz) at the shaft

Figure 4.11: *Equivalent load moments for the turbine at type "C" location*

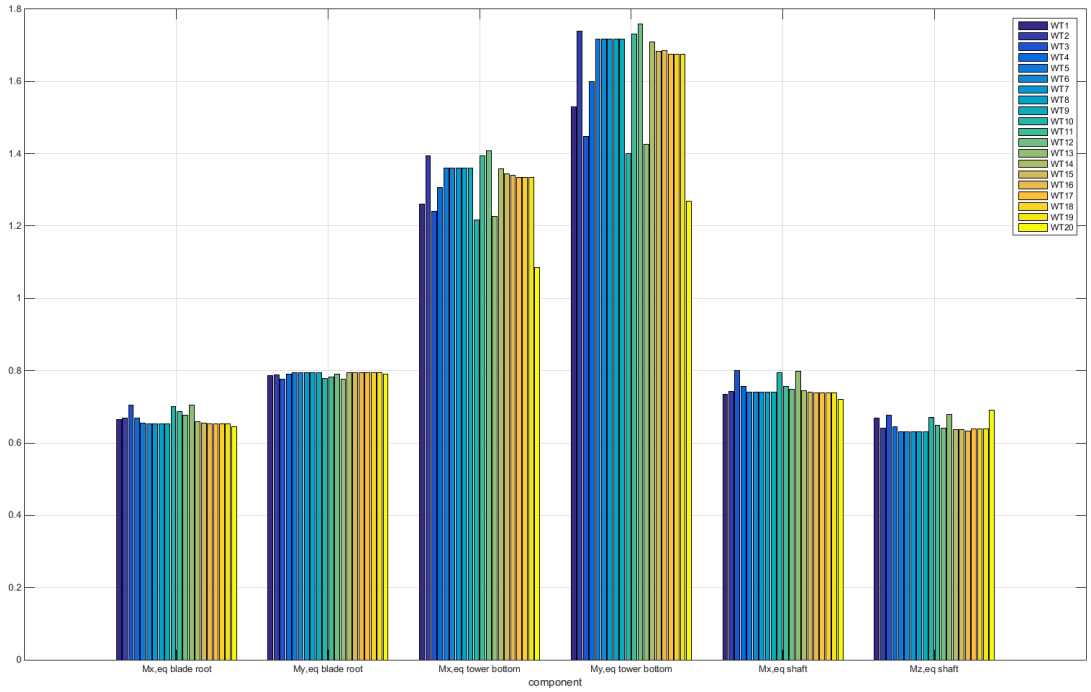


Figure 4.12: *Fatigue degradation of each component w.r.t. the turbine.*

Summarised parameters of scaled Middelgrunden wind farm is given in Table 4.2.

Table 4.2: *Scale Middelgrunden wind farm analysis summary. Price of electricity considered is 0.37€/kWh, discount rate 8%, cable price 675€/m, O&M costs account for 25% of total investment, turbine price is 1M€/MW, turbines and cabling account for 40% of total investment, interest rate of 8%.*

Parameter		
AEP	206.1	GWh
Capacity Factor	23.61	%
Loss due to the wake	16.6	%
Number of turbine	20	
Cable length	4116.7	m
Cost of turbines	100	M€
Investment cost	342.6	M€
Fatigue damage factor	4.283	
O&M costs (yearly)	18.34	M €
Profit from electricity (yearly)	76.25	M€
NPV of discounted flows	19.86	M €

4.2 Multi-stage optimisation of wind farm layouts

Engineering an optimal wind farm layout consists of determining the number of wind turbines and their respective locations so that the farm results in greatest financial value. While there may be other objectives, such as minimising the cost or maximising the efficiency, financial value (in this case NPV) as the objective function is considered to include all of these implicitly. The net present value is the financial balance between the positive and negative cash flows. While the positive cash flow comes from selling the electricity, the negative cash flows originate in the costs of maintenance and expenses of returning the loan. Engineering an optimal layout is thus done by maximising energy production while minimising both the investment and the O&M costs.

It is assumed that in case when the number of turbines is predefined, optimisation results mainly on minimising the wake effect. Knowing the investment is dominated by the cost of turbines (number of turbines), minimising the wake effect is not expected to result in high improvement. The change of funding requiered will be only due to the change in the cable length or the costs of foundations. In case the water depth does not change significantly throughout the site, only change to the investment can be done by modifying the internal cabling. Minimising the wake will, however, require higher turbine spacing which will increase the cable length and consequently its cost.

Setting the NPV as the objective function for the optimisation, while not allowing the number of turbines to vary, is not considered to be very important because the expected improvement lies only in the reduction of the wake effect. By this, it is not meant the NPV, or any other costs should not be considered. On the contrary, optimisation should be done to maximise the financial value of the farm. however, knowing the wake effect on the power production and fatigue degradation might be correlated, and keeping the turbine count constant, it might be more practical to use a simple objective function, such as capacity factor.

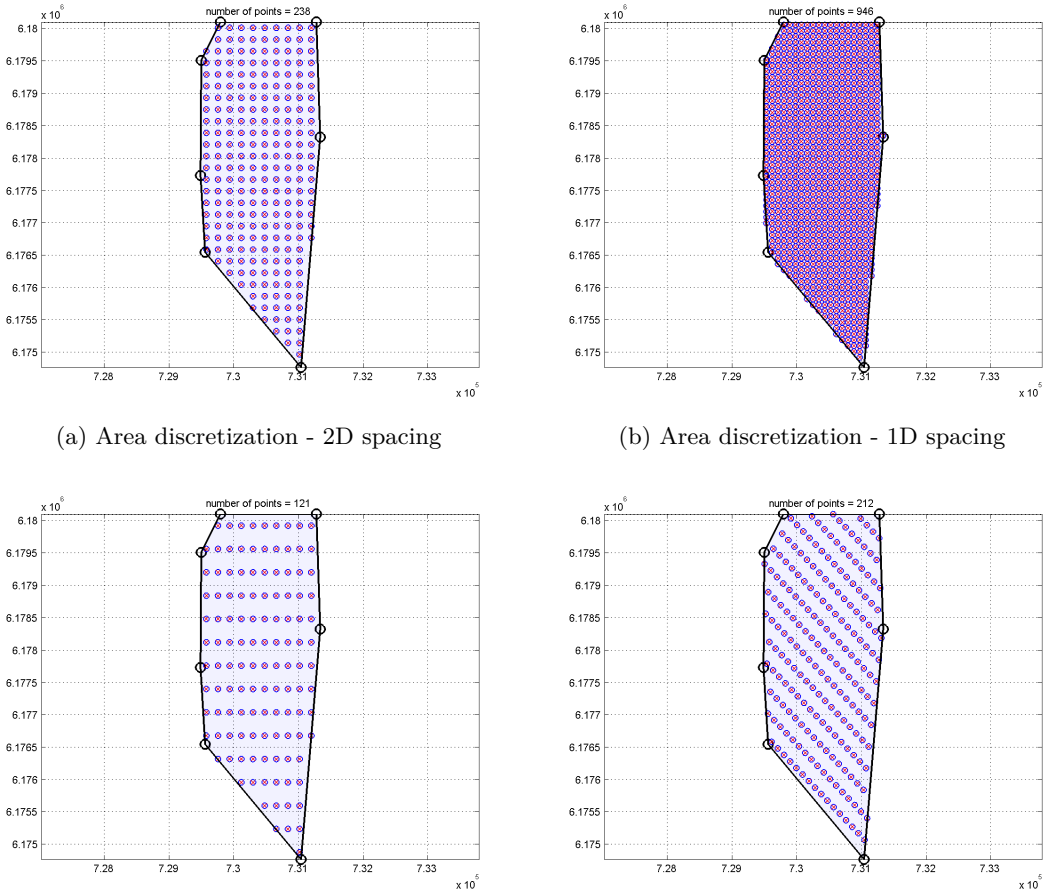
On the other hand, setting the number of turbines as a design variable will ensure greater freedom during the optimisation, and will give a new perspective of what an optimal layout should be.

Number of wind turbines is an integer variable. As such, it requires special attention if it is to be optimised using gradient search algorithm (GSA). Genetic algorithms, on the other hand, operate on a structured grid seem as an ideal solution for this issue. As a consequence, the layout optimisation is first performed with lower fidelity w.r.t. the position, but with the goal of finding the suitable amount of turbines. Upon determining the turbine count, fine tuning is done on a continuous domain by shifting the turbine

location and minimising wake effects. Because the GSA requires initial solution, and since the genetic algorithm (GA) conveniently results in numerous solutions, the optimality of the turbine count may further be confirmed by optimising individuals featuring few turbines more or less than the optimal individual.

4.2.1 Stage 1 - Genetic engineering of wind farm layouts

In the first stage, an area available for wind farm development is discretized in a number of potential wind turbine locations. These points may be spaced by any desired distance, and the grid may be angled perpendicular to prevailing wind direction(s) (example in **??**).



(a) Area discretization - 2D spacing

(b) Area discretization - 1D spacing

(c) Area discretization - variable spacing

(d) Area discretization - angled grid

Figure 4.13: *Discretization of wind farm area into possible turbine locations for the purpose of using an genetic algorithm*

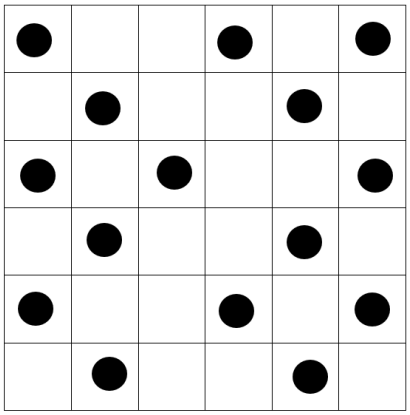
After the area is discretized into n locations, each determined by a set of X and Y coordinates, a binary value is assigned to every point. This is a gene that marks whether a turbine exists at the location (allele value of 1) or if the location is empty (allele value of 0). String assembled out of these n genes represents a genetic coding for the particular layout and thus represents an individual. Throughout the genetic optimisation, several individuals, termed a population, will be evolved through processes of crossover and

mutation until the individual with desired characteristics is found.
Due to the fact that every gene is allowed to change its allele throughout the search, genetic optimisation will implicitly take the number of turbines as a design variable while looking for an optimal layout. However, number of possible combinations in a string of n elements where each element is allowed to take two values is very large (2^n combinations). This number is slightly lowered knowing certain layouts may be excluded from the search from the very beginning. For example, the number of turbines may be bounded between the minimum, as a way of satisfying some targeted AEP, and maximum, as for example a consequence of a limited budget.
Genetic algorithms do not require computation of derivatives. The search through the genetic material is achieved by the seemingly random exchange of genes and is guided by propagating the fittest individuals, while allowing those with poor characteristics to "die-out".
Genetic algorithms are often described as a global optimisation tool. However, without going into the theory behind why genetic algorithms work, one should have in mind the only valid way of making sure the global optimum is reached, is by evaluating every possible solution (individual). Moreover, this particular individual will then be a global maximiser only for the current grid. Since there is an infinite ways of discretizing the area, the search for the global optimum makes no sense. Nevertheless, since genetic algorithms operate on a grid of finite number of points and since the theory behind their function has sound mathematical foundations [9], they represent both an easy and robust way of examining many layouts while converging towards the optimal one. Because GA implicitly consider more layouts than they actually evaluate, a fast reduction of possible solutions to a population of highly fit individuals is eminent.
The genetic algorithm used in this project is based on the Simple Genetic Algorithm presented in [9]. However, several simple adjustments were made.

Mapping, injection and permutation

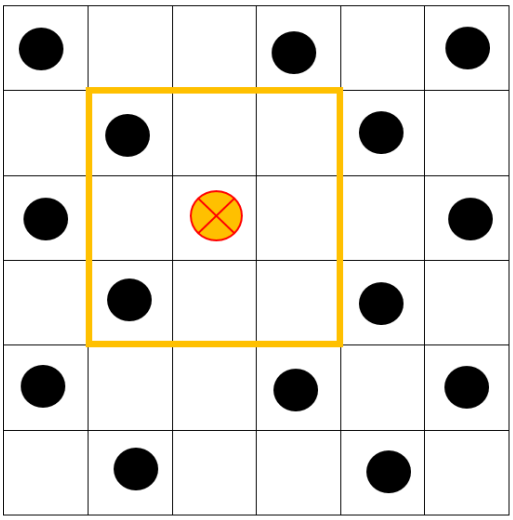
Crossover is a mechanism of exchanging genes between individuals in the population. The locations and the number of exchanged genes is chosen randomly. First, the gene is targeted and then, depending on the size of the crossover, neighbouring genes are chosen and together with the targeted gene copied to another individual. Because the individual represents the farm layout, one can reason there must be a certain relation between genes (wake effect). Due to the coding of each layout, these genes might not be next to each other. The idea is to exchange those genes which translated to a wind farm layout result in points

clustered together. Visually, the layout and its genetic coding is shown in Figure 4.14. By using the layout mapping of possible locations in the crossover phase, algorithm is able to target those points which come as neighbouring in the layout. This is shown in Figure 4.14. It is the assumption, that if an individual is highly fit, than its genes are also highly fit if examined on one-by-one basis. Said in wind farm terms, this means an optimal wind farm consist of wind turbines of optimal performance. Thus, crossing of the genes as explained means the turbine, its spacing to the neighbours in all directions and possibly few of its neighbours are exchanged as a cluster. As a consequence, the performance of the gene in the centre of the cluster is preserved. If the fitness of the cluster was low, the resulting individual will be slowed in the reproduction, if it was high, the reproduction of the individual and spreading of its fit clusters will continue.

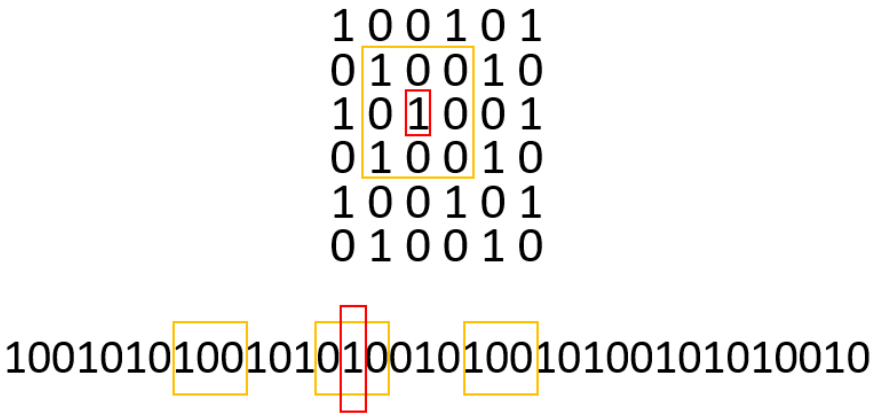


100101010010101001010010100101010010

(a) Wind farm layout given as 6by6 grid and its genetic coding



(b) The point marked is the crossing target and the square identifies its neighbours



(c) Crossing target and its neighbours identified in the genetic coding of the layout

Figure 4.14: *Gene mapping in layout coding*

As mentioned in the introduction, the number of turbines within the layout is allowed to vary between an upper and lower limit. Layouts with number of turbines outside these bounds are graded 0 in the start. As a consequence of not allowing those with fitness of 0 to reproduce, many individuals throughout the optimisation will drop out, and it may occur the population becomes over-crowed with same specimens. The layout is usually discretized in a grid with points spaced for 2 turbine diameters. Knowing the optimal layout will result in higher distance (if possible), one can expected this phenomena to occur often. The issue is solved by injection of the new genetic material, when such a

digestion occurs. Furthermore, by permuting the acceptable individuals and creating the new ones ready for injection, the number of genes having the allele of 1 in the population is kept between the desired bounds.

Mutation in genetic algorithms serves as introduction of chaos, i.e. to ensure genetic diversity throughout the search. Although the inrush of the permuted individuals contributes to this cause, a random effect is added to keep consistent with the basis of genetic optimisation. Mutation is, however, performed as a permutation of genes on a randomly chosen individual. By permuting the genes, the number of turbines remains constant, and the individual will be accepted as a valid solution.

Genetic optimisation enables examination of numerous different layouts. It does not suffer pitfalls, which may occur due to finite precision. It does not require complex calculations of function derivatives, nor does it exhibit any bias towards starting points. This makes it a very robust search strategy. Moreover, it implicitly takes the number of turbines as a design variable, and due to the fact it operates on a predefined grid, all the solutions, without exception, satisfy the spacing constraints. Genetic algorithm is furthermore pretty simple to code and if used properly may produce a valuable database of all sorts of solutions. These may subsequently be used for the sensitivity analysis as well as starting point for the optimisation algorithms of finer precision. Because they do not operate on a continuous domain, genetic algorithms cannot fine-tune the layout by positioning the turbines on other locations than those specified beforehand. This is the reason why the result of a genetic search should further be improved using for example the gradient based optimisation.

4.2.2 Stage 2 - Tuning the optimal solutions by GSA

Purpose of the genetic algorithm is to evaluate numerous layouts and indicate the optimal number of turbines, as well as their approximate positions. However, in genetic search the turbines are not allowed to move in the continuous domain. Thus, fine tuning that minimises particularly the wake effects is not solved very well. If one desires a continuous shift of turbines, a second stage using gradient based algorithm (GSA), is necessary.
GSA require the computation of at least the function gradient. The gradient gives the slope of the function at any point, which indicates in what direction should the turbine move in order to get closer to the optimum. The disadvantage of this approach is, that it requires several function evaluations before an optimal step length is reached. While this number is still much lower than the number of evaluations used in the genetic search, it

may suffer from the precision depending on the method for computing the gradient.
As explained in the beginning of this section, fine-tuning using GSA is not expected to contribute to high improvement compared to that by using GAs. Nevertheless, it may bring the already good solution to its maximum by decreasing the wake effect. As a consequence, it may happen that a sub-optimal individual from genetic search "over-takes" the optimal one after subjected to the gradient based optimisation. For this reason, several different layouts resulting from genetic search are chosen and further optimised using GSAs.
GSA used in this project is the **sqp** algorithm within MATLAB's `fmincon` function.

4.3 Middelgrunden wind farm layout optimisation

Following section studies the optimisation of the Middelgrunden wind farm layout analysed in section 4.1. Focus is first put on mechanisms within the optimisation platform, while the later part presents detailed results of finding the optimal layout.
First stage of optimisation uses genetic algorithm for searching the best individuals. Some basic settings for this purpose are listed bellow, nevertheless, important parameters will be discussed further on the example.

- The area is discretized into 238 points, each spaced for 2 turbine diameters. This makes each solution a string of 238 genes.
- The initial population consists of 20 individuals. Individuals consist of 20 genes with the allele of 1. The location of these genes is determined as a random permutation. This settings makes all initial solutions equivalent to the original layout w.r.t. the number of turbines.
- Individuals in the injection phase have turbine count within the specified bounds. For this test case it was chosen to set lower bound to 15 and upper bound to 40 turbines. Nevertheless, results of an unbounded case will be shown as well.
- Grading is based on the NPV for the lifetime of 20 years, with discount rate of 8%. The probability of reproduction is computed relative to the mean population grade. The reproduction results in a "pool" of fit candidates whose portion within the pool is proportional to their grade. This means one individual can appear in the pool several times based on its fitness and compared to the fitness of other individuals. These settings will be dealt with in the continuation.
- The crossover exchanges up to 10 genes between individuals. Mapping of the genes is done as described in the previous section.
- Mutation occurs with the probability of 5%.
- Injection is activated when more than 50% of all individuals in the population repeat in 10 subsequent steps. In that case, all repetitions are replaced by newly permuted individuals.
- Throughout the search all individuals within the specified bound are evaluated for NPV, AEP, fatigue degradation, capacity factor and cable length. A database with these values is created and filled in each new iteration. Every new individual is

also checked with the database before being evaluated. A databased is later used to examine the sensitivity and trends in the convergence towards optimal layout. Furthermore, individuals carrying certain characteristics may be extracted from the database and further optimised using GSA.

- The stopping criteria for the genetic algorithm is discussed in the continuation.
- The attribute of the optimisation is the offset from the initial layout.
- The objective of the optimisation is the net present value computed as in Equation 1.22.

The aim of this section is to examine and study several questions:

1. What is the benefit of using combined optimisation strategy consisting of genetic algorithm first and gradient search algorithm second?
2. On what level and in which direction is improvement expected in each of the optimisation phases?
3. How to determine the stopping criteria for genetic algorithm?
4. How does the objective function behave w.r.t. number of turbines?
5. Could the bounds on the turbine count be approximated as to shrink the design space?

4.3.1 Gradient search algorithm in practice

Optimisation of Middelgrunden wind farm layout using just GSA is discussed next.
The number of turbines in this case has to be set a priori and will remain constant throughout the optimisation. This is due to the fact GSA operates on continuous domain, while the number of turbines has an integer value. As a consequence, investment is not expected to change and the improvement is expected as a result of reducing the wake effect (higher capacity factor, lower wake loss and less fatigue damage). Although the length of the cable may change the investment, knowing the wake effect reduce by spreading turbines further away, the cable length is expected to increase.
The attribute (design variable) of the optimisation is the offset from the initial turbine positions. The bounds on the movement of each turbine have been defined w.r.t. the distance of each turbine to the area borders. Note this makes sense only for the initial starting points while during the optimisation, each turbine is constrained within the area

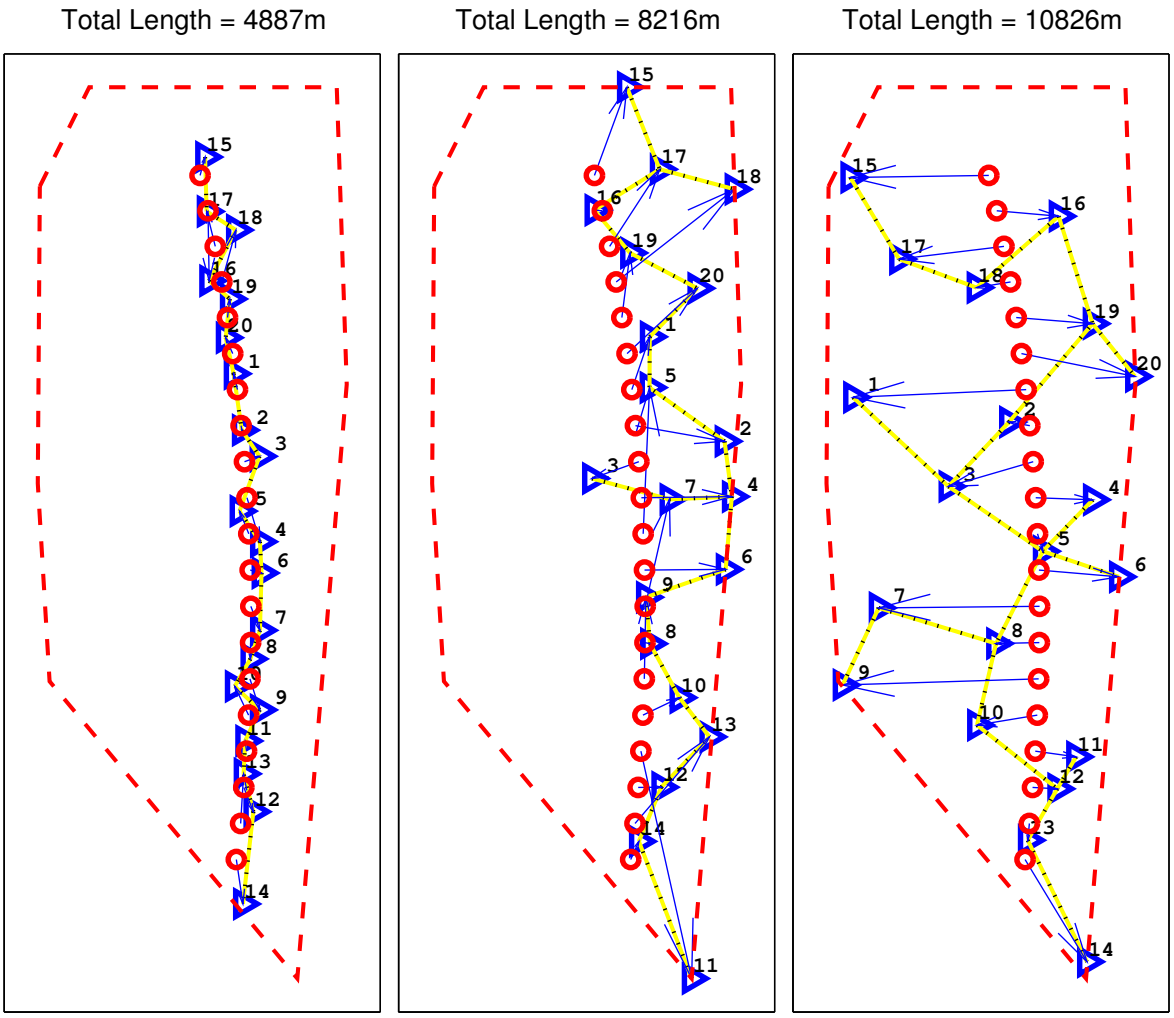
borders and to a minimal distance of 2D towards neighbouring turbines. In order to trigger the optimisation, the starting turbine locations have been chosen as a random offset from the original position. Since the original Middelgrunden layout is shaped as a curve stretching in Y-direction, the offset is emphasised in X-direction.

The cases studying starting points for GSA are shown in the following.

For the first case the initial offset in X-direction was allowed up to 1 turbine diameters (labelled 1st solution), for the second case up to 2.5 (labelled 2nd solution), and for the third case up to 5 turbine diameters (labelled 3rd solution). Table 4.3 gives the summary of important parameters, and Figure 4.15 shows the resulting layouts.

As seen in Table 4.3, the assumption regarding the improvement is confirmed. Optimised layout have been found by reducing the wake effect on account of increased spacing between turbines. As a consequence, the AEP is increased and fatigue degradation lowered. The cable length is increased as expected. In the 3rd case it is increased more than 160%! Nevertheless, the NPV manages to improve sufficiently. Results comparing wake loss of GSA optimised layouts w.r.t. different starting points are shown in Figure 4.17, and Figure 4.16 shows relative change in fatigue damage. Improvement in NPV is rather high. This might be an indication the weighting of costs in the computation of NPV, or in particular the investment, might require some tuning. On the other hand, from Figure 4.15 it can be noticed that the change in the turbine positions is done by shifting the turbines outside the original "column" ordering. Knowing the prevailing wind comes from 180°, it is understandable the turbines will capture more energy this way. As a consequence NPV increases significantly.

Notice further that setting larger offset from the initial layout results in a layout less like the original one and moreover, increases the improvement. For small initial offsets, the optimisation gets stuck in the local optima and does not manage to further improve the objective. For this reason it is important to examine several starting layouts. Genetic optimisation will prove to be rather useful for this purpose since, as explained, it provides a vast database of possible solutions. The genetic optimisation has no defect of being bias towards the initial guess, and this is one of the reasons why it is combined with GSA.

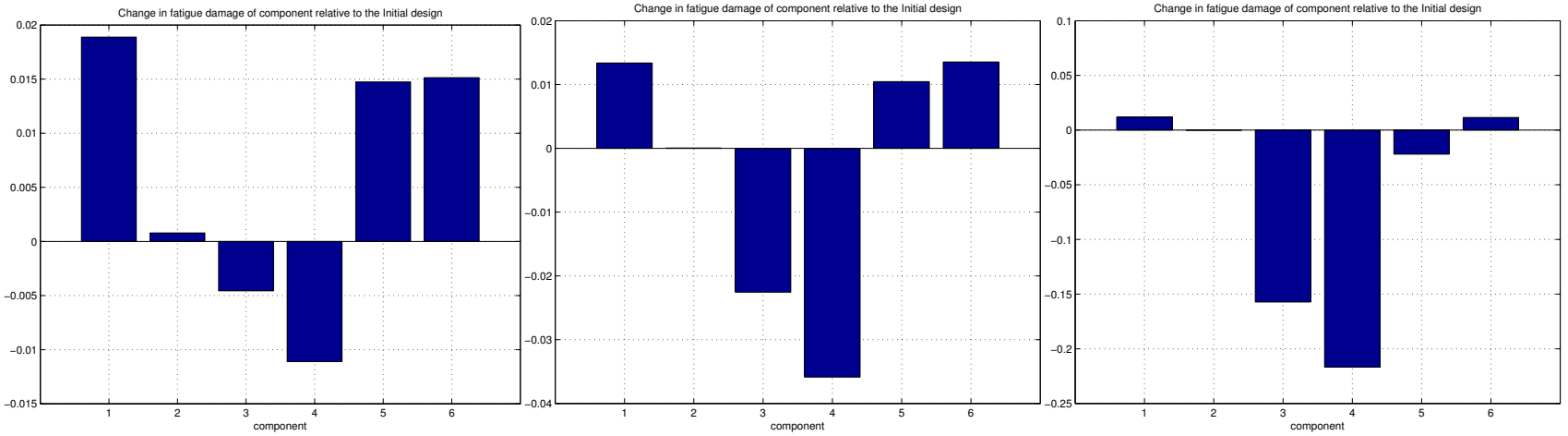


(a) 1st solution - initial offset up to 1D (b) 2nd solution- initial offset up to 2.5D (c) 3rd solution- initial offset up to 5D

Figure 4.15: *Gradient search optimisation of initial Middelgrunden layout. Study of 3 different initial solutions*

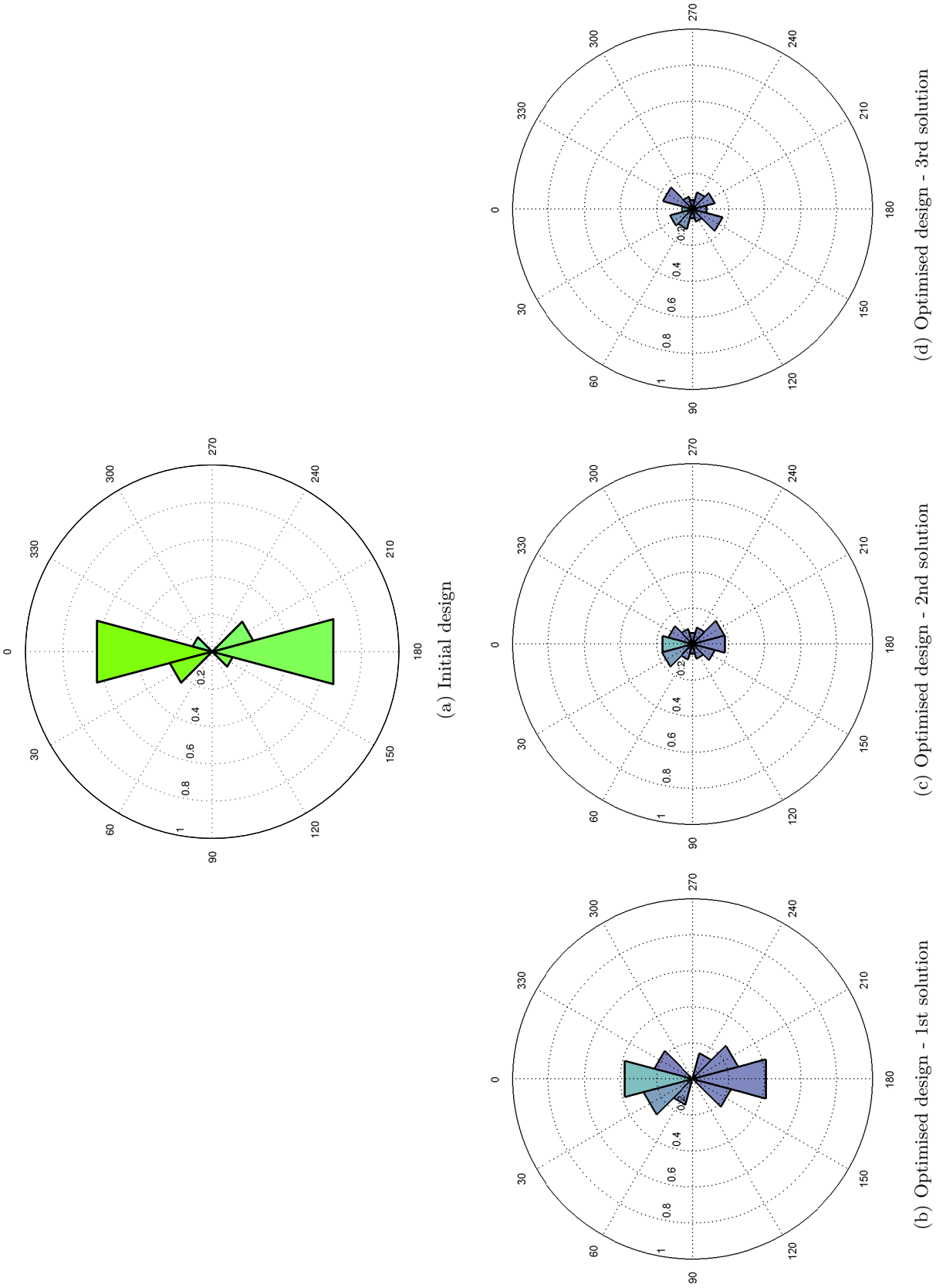
Table 4.3: *Results of GSA optimisation on the initial layout*

	AEP	capacity factor	fatigue degradation	cable length	number of turbines	wake loss	NPV
INITIAL layout	206.11	0.236	4.28	4116.72	20.00	0.166	1.986E+08
post-GSA - 1st solution	212.11	0.243	4.27	4887.50	20.00	0.174	2.181E+08
Rel. change - 1st solution	2.91%	2.914%	-0.27%	18.72%	0	4.738%	9.82%
post-GSA - 2nd solution	226.26	0.259	3.91	8215.71	20.00	0.122	2.731E+08
Rel. change - 2nd solution	9.78%	9.779%	-8.68%	99.57%	0	-26.643%	37.48%
post-GSA - 3rd solution	234.05	0.268	3.73	10826.44	20.00	0.101	3.003E+08
Rel. change - 3rd solution	13.56%	13.558%	-12.95%	162.99%	0	-39.543%	51.16%



(a) Optimised design - 1st solution (b) Optimised design - 2nd solution (c) Optimised design - 3rd solution

Figure 4.16: *Fatigue damage of different components. 1 - eqv. load (Mx) at the blade root, 2 - eqv. load (My) at the blade root, 3 - eqv. load (Mx) at the tower bottom, 4 - eqv. load (My) at the tower bottom, 5 - equivalent load (Mx) at the shaft, 6 - equivalent load (Mz) at the shaft*



(a) Initial design

(b) Optimised design - 1st solution

(c) Optimised design - 2nd solution

(d) Optimised design - 3rd solution

Figure 4.17: *Wake loss per sector*

Knowing the optimisation is directed towards reducing the wake effect, one may also wonder if it is necessary to include fatigue degradation in the objective, in the other words, would parameter such as wake loos or capacity factor suffice. To examine this, one should look at the sensitivity of the objective function, which is done as a part of the following section.

4.3.2 Sensitivity of the objective function

It is the assumption that increasing the amount of turbines in the layout of a wind farm has a limit, after which the increase in AEP cannot be justified by high investment. Moreover, if the layout cannot be optimised to reduce the effects of the wake sufficiently, cost of maintenance will surpass the positive cash flow. The farm in that case becomes saturated and no longer profitable.
In this section, the objective function will be examined for its sensitivity to several parameters. The aim is further to approximate the bounds on the turbine count as to predict when the wake effect becomes too intense for efficient power production. For the purpose of analysing the sensitivity a database of individuals created during genetic search is used.
Four main parameters are considered for the sensitivity:

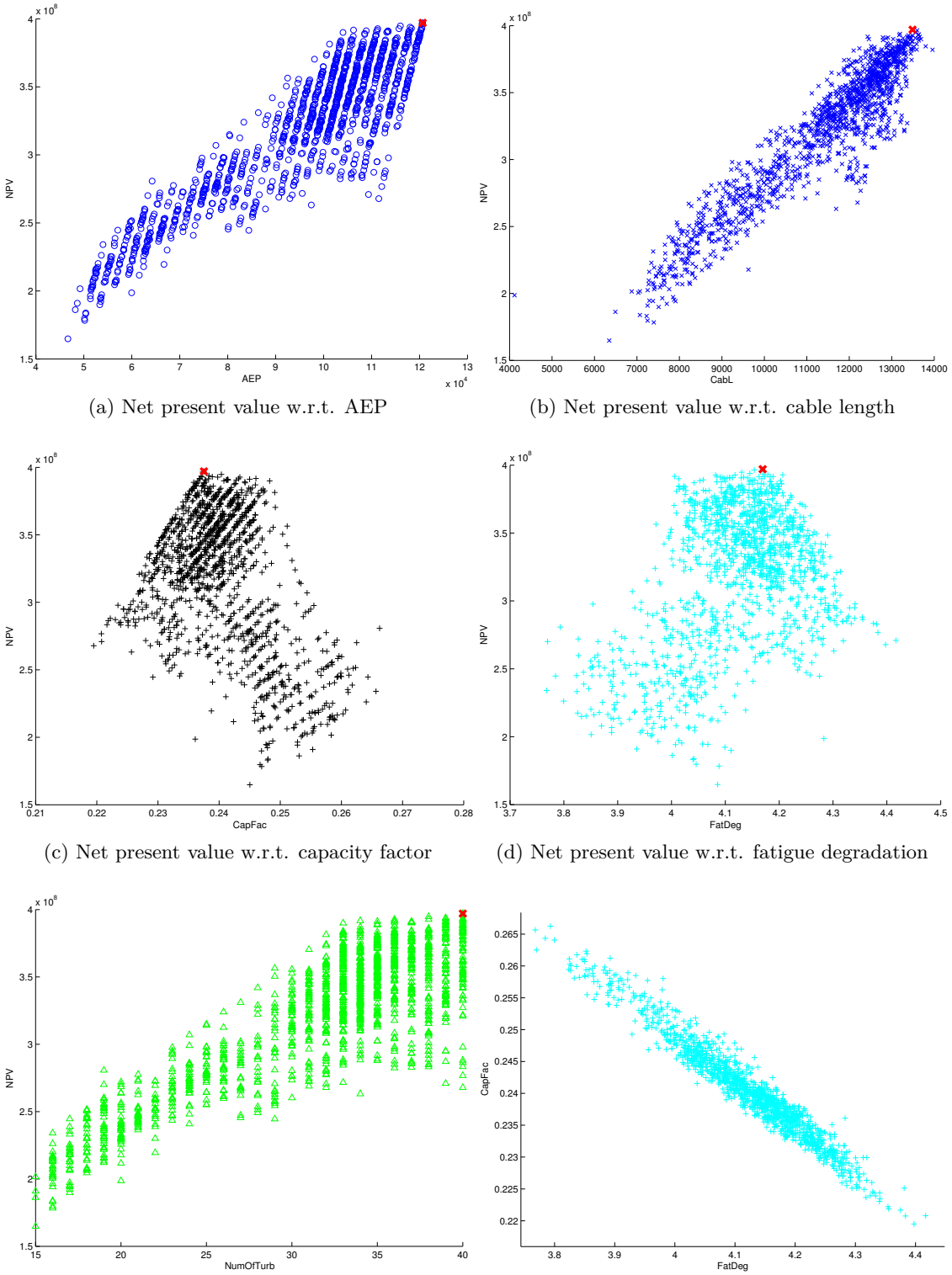
- Number of turbines
- Cable Length
- AEP
- Fatigue degradation

Following facts may be reasoned immediately. Increasing the number of turbines increases both AEP, and the investment. It will also increase the wake effect which will as a consequence increase the costs of maintenance and cause less efficient production.
The cable length has small effect to the investment (in the order of 5%). However, note the cabling is already optimised to result in the minimal length for given layout.
The AEP is the positive cash flow, thus it is crucial for it to be sufficiently high. It is also important for a farm to produce energy in an efficient way, thus minimising the damage due to fatigue. The cost of maintenance is in the order of 25% of the investment. The fatigue degradation factor is allows this to be exceeded.
The dependency of NPV to the mentioned parameters is not straight forward. Computing the NPV as objective does not include the position of the turbines as explicit input. On

the contrary, prior to the calculation of NPV, the cabling must be determined in a separate function, and the relevant wake sources are determined as a function of spacing and wind direction. Consequently, NPV is not a smooth function. The sensitivity analysis is thus made using the scatter plots shown in Figure 4.18. The scatter plot showing dependency of NPV (objective function) w.r.t. AEP is shown in Figure 4.18a, w.r.t. the cable length in Figure 4.18b, capacity factor in Figure 4.18c, fatigue degradation in Figure 4.18d and number of turbines in Figure 4.18e.
Notice from Figure 4.18e the NPV increases fast with number of turbines up until 33. Same trend is visible for the AEP in Figure 4.18a. Notice the optimal layout (marked with red cross) is the one with the highest AEP. This indicates the objective is influenced the most by the positive cash flow, i.e. production. On the other hand, it might indicate the profit from electricity is weighed as too valuable compared to other costs. In case optimisation tool such as this is to be put in practice for a "real-life" project, the weights, i.e. the prices, should be bench-marked.
The cable length increases w.r.t. the increase in NPV (Figure 4.18b). This was expected in the beginning due to the tendency of increasing the spacing between turbines.
Examining scatter plots for capacity factor and fatigue degradation w.r.t. NPV (Figure 4.18c,d) it is hard to identify any trends. However, there is resemblance in those two plots. First, notice the optimal layout is not the best results w.r.t. those two parameters. This is expected knowing the increase of production has to happen on the account of higher wake effect. Second, from Figure 4.18f one can notice the relation between capacity factor and fatigue degradation. The dependency is linear for examined solutions. This is an indication the wake equally affects the losses and the fatigue damage. Thus the question arises if is it necessary then to include the fatigue into the objective and could it be possible to express fatigue degradation as a function of the capacity factor (or some other wake related parameter). Furthermore, wouldn't the optimisation with the objective of decreasing the wake effect by increasing the capacity factor have implicit effect of reducing the fatigue as well? To answer take the following into account:

- Capacity factor is implicitly maximised only in the second stage of optimisation, after the number of turbines has already been determined. It cannot be used as a measure of estimating the turbine count, since is degrades for any increase in the amount of turbines.
- Computing NPV as an objective function does not consider capacity factor as input. Removing the fatigue degradation from objective will thus require a new model, which would "translate" the capacity factor, or some other measure of the wake effect, to costs.

- The meandering of the wake manifests as cascades of altered wind conditions hitting the turbine. These cascades are main reasons for increase in fatigue damage but may not cause significant loss in production.



(a) Net present value w.r.t. AEP

(b) Net present value w.r.t. cable length

(c) Net present value w.r.t. capacity factor

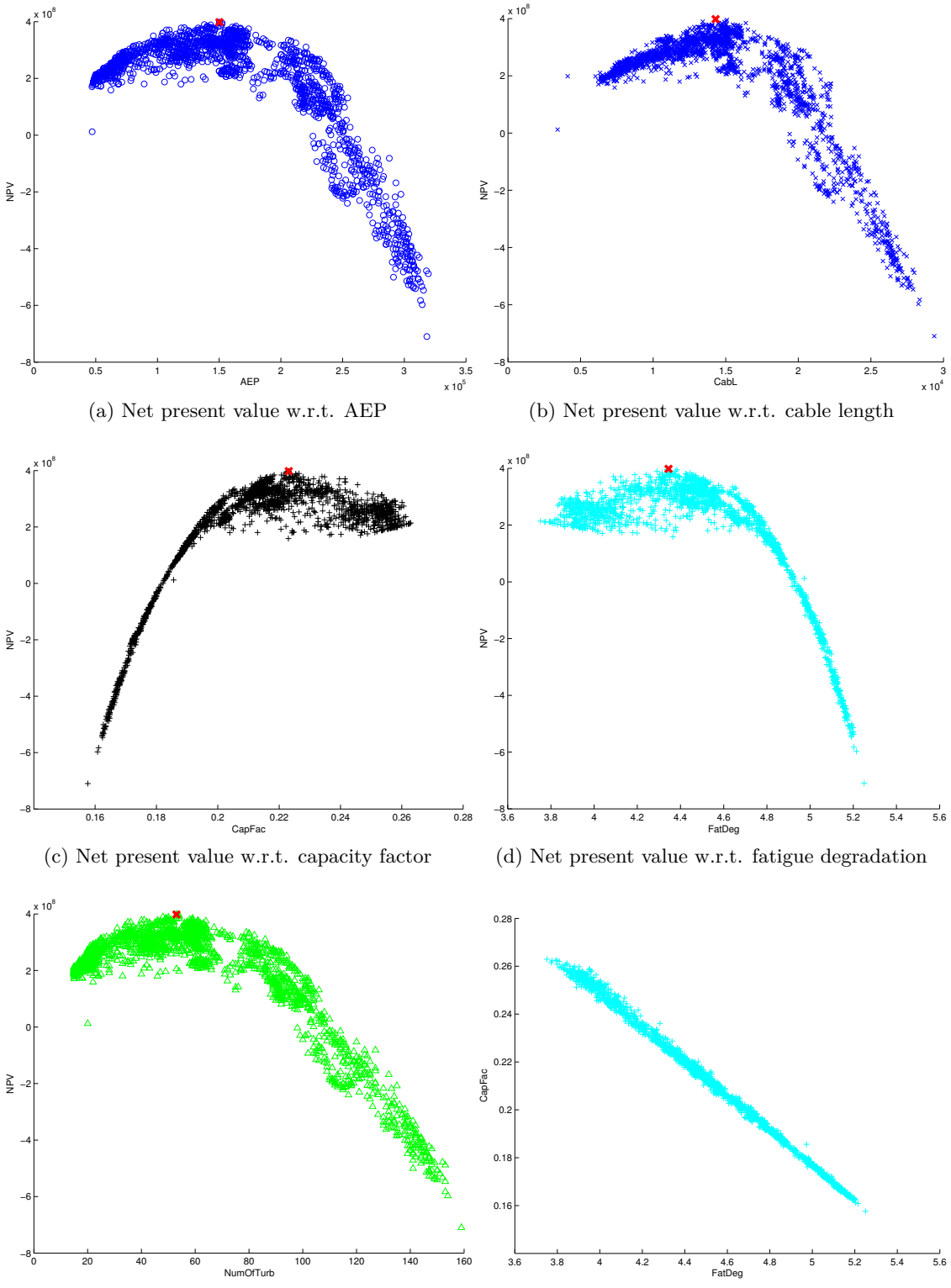
(d) Net present value w.r.t. fatigue degradation

(e) Net present value w.r.t. number of turbines

(f) Fatigue degradation w.r.t. number of turbines

Figure 4.18: *Sensitivity of NPV to other characteristics of the wind farm. Individuals evaluated during the genetic search. The optimal solution is marked as red ×. Turbine count bounded between 15 and 40.*

Due to these facts, fatigue analysis remains necessary. However, a study should be made in the correlation of wake effect to power loss and fatigue damage.
Though some patterns were identified in the scatter plots shown in Figure 4.18, the threshold identifying point were the investment, or costs of maintenance, becomes dominant is not visible. To find this "breaking point", the upper bound on the turbine count has been removed, and another test was performed. Resulting scatter plots are shown in Figure 4.19. Notice the threshold clearly exists and is identified around 50 turbines. Figure 4.19 also shows the optimal layout was not found in the previous example since the NPV continues to rise to it maximum at 53 turbines. Apart from starting to decrease, notice also the NPV crosses 0 after certain number of turbines has been introduced to the farm.



(a) Net present value w.r.t. AEP

(b) Net present value w.r.t. cable length

(c) Net present value w.r.t. capacity factor

(d) Net present value w.r.t. fatigue degradation

(e) Net present value w.r.t. number of turbines

(f) Fatigue degradation w.r.t. number of turbines

Figure 4.19: *Sensitivity of NPV to other characteristics of the wind farm. Individuals evaluated during the genetic search. The optimal solution is marked as red ×. Upper bound on the turbine count has been removed.*

4.3.3 Design space

This section analyses the number of possible solutions in the genetic search. For an area discretized in n points the reasoning is as follows. If no bounds are set on the turbine count, then each of the genes (bits) in the individual (string) can be either 0 or 1. The number of possibilities is thus the product of number of values being chosen for each of the elements $2 \cdot 2 \cdot ...$ (n times) $= 2^n$. If we set the bounds on the turbine count, we may think of the situation as having r turbines to position on n possible locations. Thus, for the 1st turbine there are n possible locations, for the 2nd turbine $n-1$ possible locations, third $n-2$ and so on. The number of solutions would thus be $n \cdot (n-1) \cdot (n-2) \cdot ..(n-r) = \frac{n!}{(n-r)!}$. However, since it does not matter in which order we are choosing the locations, this number has to be reduced for the number of repeating combinations. The result is $\frac{n!}{r!(n-r)!}$ for r turbines. Setting the bounds on the turbine count the number of possibilities needs to be summed over each number of turbines $\sum_i \frac{n!}{r_i!(n-r_i)!}$.

4.3.4 Genetics behind convergence

The genetic algorithm has been tested for 200 non-stop iterations in order to understand the following issues:

- How large should the population be?
- On which basis should the individuals be chosen for the reproduction?
- What should the stopping criteria be?

Reasons why careful consideration is given to these matters is explained in the following. A "traditional" method for computing the probability of an individual appearing in the reproduction pool is by dividing the fitness of an individual by the mean population fitness. However, in cases where small percentage of individuals survives crossover or mutation, and especially in case when the population size is small, it is likely that such grading causes problems in terms of convergence. Figure 4.20 shows the history of the mean population fitness throughout the 200 iteration. The red lines mark the iterations where the injection of new individuals occurred as a consequence of having 50% duplicates in the population. In 200 iteration, a total of 1963 individuals has been evaluated. Knowing the population size for this test was set to 20, it is clear that something is not allowing the individuals to alter and evolve throughout the optimisation. Although the algorithm works fine and manages to find the layout of better performance, it operates more as a random search and less like a genetic algorithm. Thus, its convergence towards the optimum is presumably

slower. One of the reasons for having too many similar individuals is certainly exclusion of layouts featuring number of turbines outside the set bounds. Keep in mind that bounding the turbine count between 15 and 50 for 238 grid points equals less than $2,65 \cdot 10^{-18}\%$ of possible layouts! In case a layout is graded 0, is is immediately excluded from the reproduction pool, and when a large amount of such samples exist in the population, there is more chance that the same individuals will be crossed. As a consequence, the exchange of genes is not producing unique individuals. While this problem was expected, and as a consequence, injection mechanism was introduced, it was not expected its effect will be in such a significant scale. The second reason for the "stale genetic material" might lie in the fact the population is too small or that, furthermore, the amount of samples in the reproduction pool is too few.

The grading of the individuals for the purpose of creating the reproduction pool is one of the crucial settings in genetic algorithms. The grading in this case refers to the translation of the fitness into a probability of an individual being reproduced and crossed. For example, if the grading is "too strict" it will not allow less fit layouts to reproduce. On the other hand, if the grading does not register small changes in the fitness of individuals, the algorithm will lose its "genetic" properties. According to [9], at the start of the search, the individuals in the population are random, and there might be few that stand out. It this case the selection rule as in Equation 4.1 will propagate the extraordinary individuals and cause premature convergence. In the later stage of the search, the population is less diverse, thus, the difference between the population mean and maximum will be small. As a consequence, the good and the less good samples will be treated equally. While this ensures the reproduction is diverse, we would like to achieve convergence towards the best individual and not exchange gens forever.

For the reasons mentioned above, the population size has been increased to 30, the fitness is scaled linearly as shown in Equation 4.2 and Equation 4.3 [9], and the reproduction pool size has been increased. Note the number of samples in the reproduction pool is still determined using the expression in Equation 4.1 and multiplied by the constant (in this case 10) to scale the amount of samples in the reproduction pool. Furthermore, since the fitness is allowed to be negative (NPV), the grade of all samples is first increased by the minimal fitness in that population.

Fitness scaling and percentage of samples in the reproduction pool is shown in Table 4.4 and Figure 4.21 for case where few individuals dominate the population, and in Table 4.5 and Figure 4.22 for the case when individual fitness does not vary far from the mean population fitness.

The convergence history of the algorithm employing linear fitness scaling is given in

Figure 4.23.

Based on Figure 4.23b, a decision regarding the stopping criteria is made. The genetic algorithm will terminate in cases w.r.t. the following criteria:

1. iteration count exceeds minimum of 150 (necessary condition)
2. relative improvement exceeds 80%
3. if no change to the relative improvement happens in 30 subsequent iterations

$$\begin{aligned} prob_i &= \frac{f_i}{F} \\ count_i &= \frac{f_i}{\overline{f}} \cdot 10 \end{aligned} \tag{4.1}$$

where

$prob_i$ is the probability of the i-th sample appearing in the reproduction pool

$count_i$ is the expected count of the i-th sample in the reproduction pool

f_i is the fitness of the i-th sample

F is the sum of population fitness

$\overline{f}$ is the mean population fitness

N is the mean population fitness

$$\begin{aligned} &\textbf{if} \quad \min(f_i) > (C \cdot \overline{f} - \max(f)) \\ &\qquad \delta = \max(f_i) - \overline{f} \\ &\qquad a = (C-1) \cdot \frac{\overline{f}}{\delta} \\ &\qquad b = \frac{\overline{f} \cdot (\max(f_i) - C \cdot \overline{f})}{\delta} \\ &\textbf{else} \\ &\qquad \delta = \overline{f} - \min(f_i) \\ &\qquad a = \frac{\overline{f}}{\delta} \\ &\qquad b = \min(f_i) \cdot \frac{\overline{f}}{\delta} \end{aligned} \tag{4.2}$$

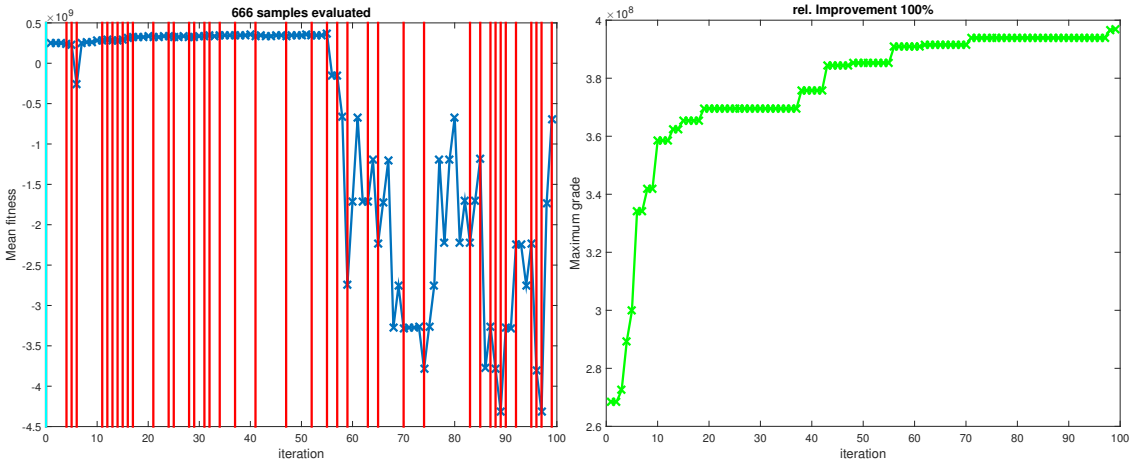
where

C is an constant between 1.2 and 2 [9]

a is the slope

b is the intercept

$$f_{scaled} = a \cdot f_{raw} + b \tag{4.3}$$

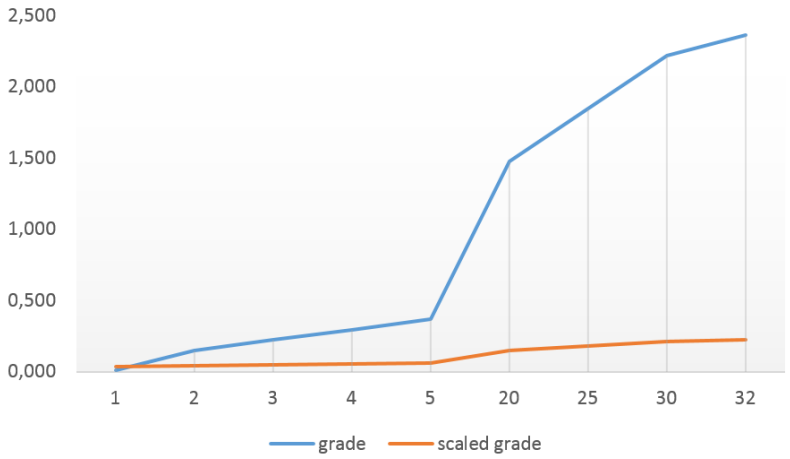


(a) Mean population fitness (b) Best individual fitness

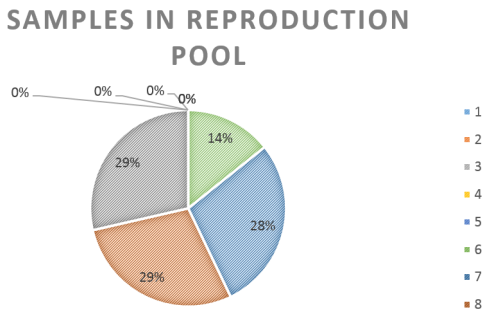
Figure 4.20: *Convergence of genetic algorithm towards optimal layout. Fitness is not scaled. Red lines indicate iterations when injection occurred.*

Table 4.4: *Fitness scaling for genetic optimisation. Case when few individuals in the population stand out.*

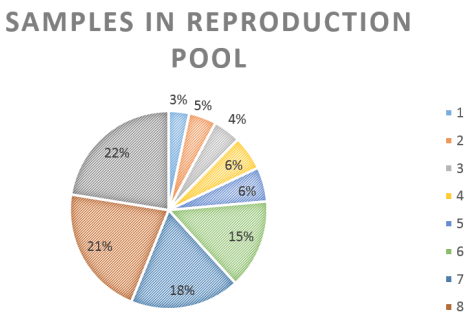
fitness	grade	samples in pool	scaled grade	samples in pool
1	0,008	0	0,035	3
2	0,148	0	0,041	4
3	0,221	0	0,048	4
4	0,295	0	0,054	5
5	0,369	0	0,060	5
20	1,475	10	0,150	13
25	1,844	20	0,180	16
30	2,213	20	0,210	19
32	2,361	20	0,222	20



(a) Comparison of fitness and scaled fitness



(b) Percentage of samples in reproduction pool using unscaled fitness

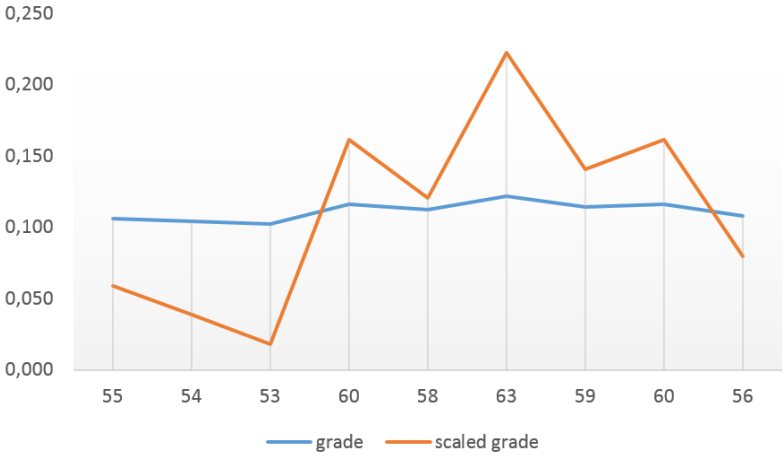


(c) Percentage of samples in reproduction pool using scaled fitness

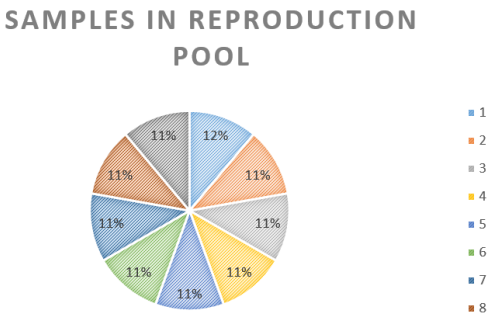
Figure 4.21: *Scaling fitness of individuals in population and reproduction pool. Few individuals in the population stand out.*

Table 4.5: *Fitness scaling for genetic optimisation. Case when individual fitness does not differ much from the mean population fitness.*

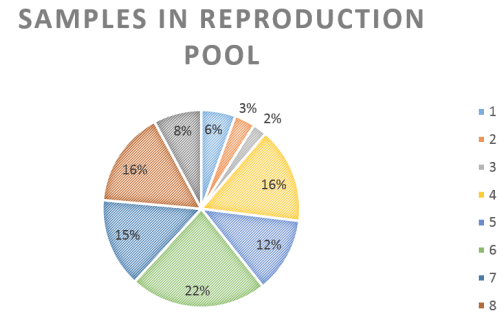
fitness	grade	samples in pool	scaled grade	samples in pool
55	0,106	10	0,059	5
54	0,104	10	0,039	3
53	0,102	10	0,018	2
60	0,116	10	0,161	14
58	0,112	10	0,120	11
63	0,122	10	0,222	20
59	0,114	10	0,141	13
60	0,116	10	0,161	14
56	0,108	10	0,079	7



(a) Comparison of fitness and scaled fitness

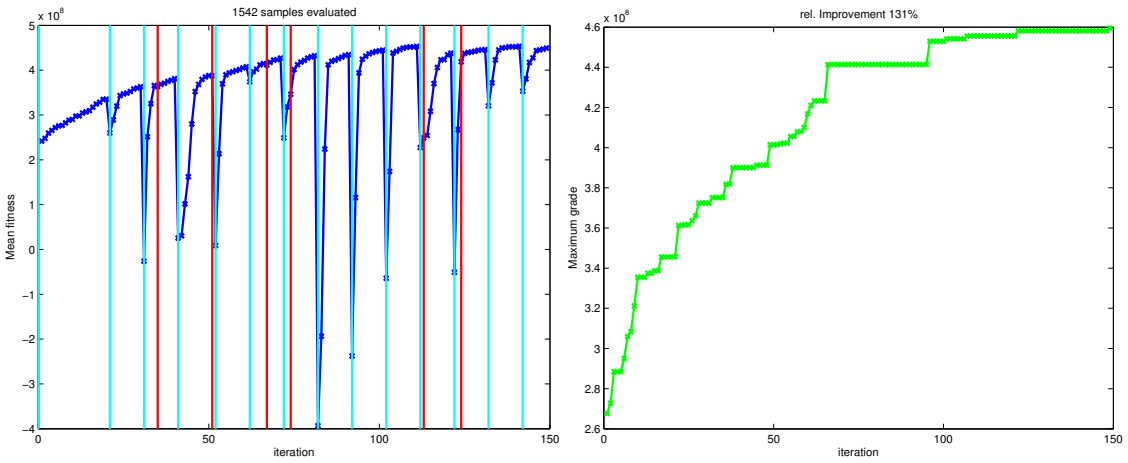


(b) Percentage of samples in reproduction pool using unscaled fitness



(c) Percentage of samples in reproduction pool using scaled fitness

Figure 4.22: *Scaling fitness of individuals in population and reproduction pool. Individual fitness does not differ much from the mean population fitness*



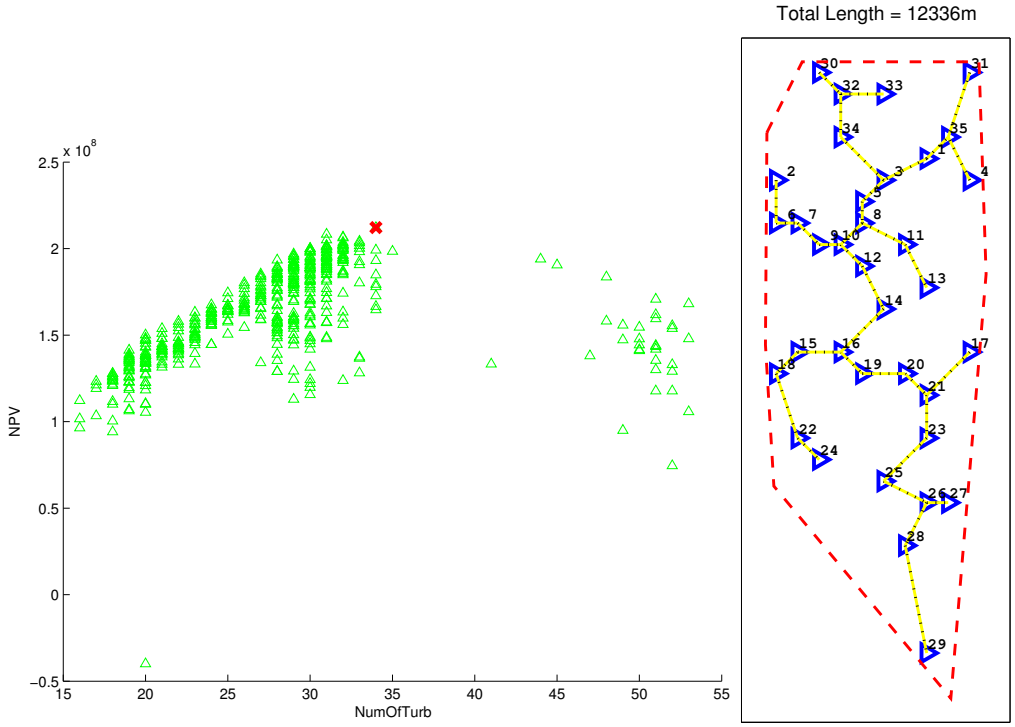
(a) Mean population fitness

(b) Best individual fitness

Figure 4.23: *Convergence of genetic algorithm towards optimal layout. Scaled fitness. Cyan lines indicate iteration where injection occurred, red lines indicate mutation.*

4.3.5 Optimisation results

The objective of the genetic optimisation is set to maximise the NPV (net present value) of the wind farm. By data-mining the database created during the genetic search several individuals w.r.t. different criteria are chosen to be initial solutions for 2nd-stage GSA. Table 4.6 lists several individuals characteristic from various perspectives. All of these layouts may serve as initial layouts for the gradient based optimisation. Mind that the global optimum for NPV may not be an optimal solution for the real-life project. On the contrary, layout may be constrained by the maximal investment or some other requirement. Furthermore, keep in mind the objective highly depends on the weighting of related costs. If, for example, the cost of cabling would be increased several times, the optimal layout would result in 35 turbines as shown in Figure 4.24.
Results of GSA optimisation on the top three individuals found in the genetic search are presented next. The results for the 2nd and 3rd best are shown in Table 4.8 and Table 4.9, respectively. The optimal layout is, however, found by tuning the GA-best individual.



(a) Scatter of NPV for various number of turbines (b) Best individual

Figure 4.24: *Result of optimisation with amplified cost of cabling.*

Table 4.6: *Fittest individuals in the genetic search w.r.t. different criteria*

criteria	AEP	cable length	fatigue degradation	number of turbines	capacity factor	NPV
INITIAL design	206.105	4116.7	4.283	20	0.236	1.99E+08
BEST	516.19	14301.3	4.345	53	0.223	3.982E+08
max Cap. Fact.	172.13	7532.7	3.752	15	0.263	2.141E+08
min cable l.	162.01	3420.0	4.973	20	0.186	1.208E+07
min fatigue deg.	242.44	9640.9	3.943	22	0.252	2.734E+08
2nd BEST	516.05	15076.5	4.338	53	0.223	3.957E+08
3rd BEST	524.02	14301.3	4.384	54	0.222	3.954E+08
4th BEST	523.15	15045.7	4.378	54	0.222	3.901E+08
5th BEST	474.44	13420.5	4.291	48	0.226	3.897E+08
BEST for 50 turb	491.93	14909.6	4.345	50	0.225	3.885E+08
BEST for 51 turb	497.58	15016.2	4.375	51	0.224	3.786E+08
BEST for 52 turb	503.19	14067.2	4.391	52	0.222	3.736E+08

Table 4.7: *Results of GSA optimisation on the fittest individual*

	AEP	capacity factor	fatigue degradation	cable length	number of turbines	wake loss	NPV
After GSA	536.02	0.232	4.33	16058.91	53	0.222	4.657E+08
Relative change	3.84%	3.842%	-0.36%	12.29%	0%	-5.15%	16.93%
TOTAL rel. change	160%	-2%	1%	290%	165%	33%	134%

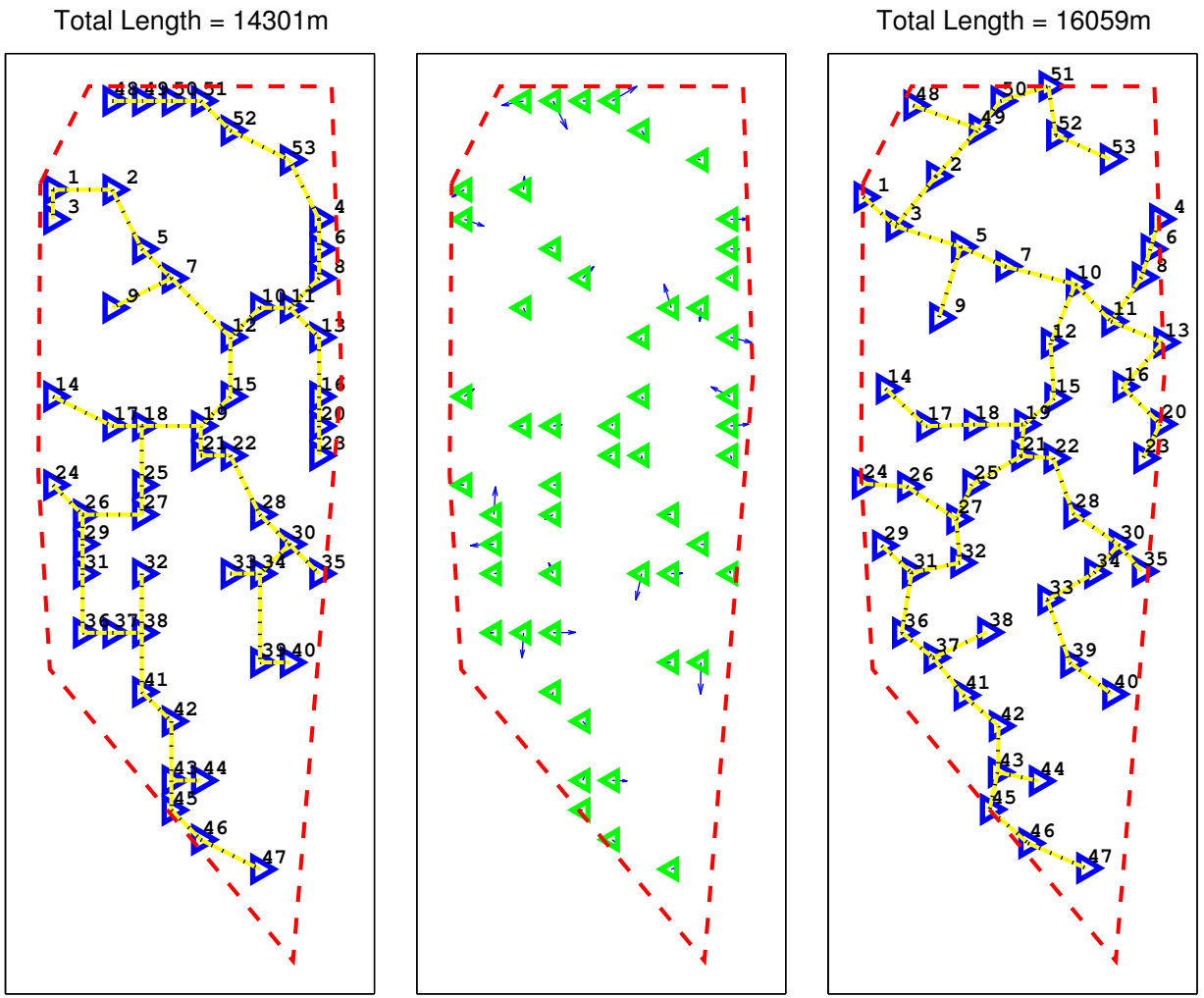
Table 4.8: *Results of GSA optimisation on the 2nd best individual*

	AEP	capacity factor	fatigue degradation	cable length	number of turbines	wake loss	NPV
After GSA	516.54	0.223	4.37	14842.45	53	0.243	3.948E+08
Relative change	0.10%	0.097%	0.73%	-1.55%	0%	0.925%	-0.23%
TOTAL rel. change	151%	-5%	2%	261%	165%	046%	99%

Table 4.9: *Results of GSA optimisation on the 3rd best individual*

	AEP	capacity factor	fatigue degradation	cable length	number of turbines	wake loss	NPV
After GSA	542.15	0.230	4.34	16076.31	54	0.222	4.602E+08
Relative change	3.46%	3.461%	-1.07%	12.41%	0%	-5.986%	16.40%
TOTAL rel. change	163%	-3%	1%	291%	170%	34%	132%

Following results compare the best layout found through genetic search (Figure 4.25a) and the layout tuned further using gradient based optimisation (Figure 4.25c).
In comparison with the starting layout (scaled Middelgrunden layout) the number of turbines increased from 20 to 53. This increased the AEP by 150%. At the same time, the fatigue damage increased from factor 4,283 to factor 4.345. Note this is the value averaged over all turbines. Capacity factor decreased to 22.3%. Nevertheless, the NPV increased by $\sim$ 100%. The resulting layout was achieved by spreading out the turbines over the allowed area, and as a consequence the cable increased to the length of 14.3km from the original 4.12km. Nevertheless, the price of the cable did not play a significant role compared to the accomplished improvement.
Further tuning using GSA brings additional increase of 16% in the NPV. This is achieved by further increasing the spacings between turbines. In turn, the capacity factor is slightly increased and fatigue damage decreased. Change in these parameters after the GSA are given in Table 4.7. The shift from the starting layout is indicated in Figure 4.25b, and the final layout is shown in Figure 4.25c. Notice the GSA optimisation has a tendency of breaking-up the clusters of turbines spaced close together.
Assuming the genetic search finds the layout to the precision of 2D (grid spacing was set as 2D), the GSA optimisation may be bounded to maximal shift within the circle of 2D radius. Though the freedom of the optimisation will be reduced, if the genetic algorithm operated as planed, the result should be an optimal solution. To stay on the safe side, however, as well as to test the precision of genetic algorithm, the allowable shift was bounded to 5D. The shift did not surpass 2.2D.
Commenting on the wake effect within the farm, the resulting layout has, as a consequence of having more turbines, higher loss due to the wake. However, as seen in Figure 4.28, the losses are not emphasised in two sectors as in the case of the initial Middelgrunded layout, but are distributed more evenly over turbines and each sectors.
Finally, notice the change in costs shown in Figure 4.29. Compared to the starting wind farm, the optimised park will require 168% higher investment. Moreover, analysing every parameter indicated in Table 4.7, apart from AEP, the resulting optimal layout performs worse than the starting point. However, it supplies 134% higher NPV, and this makes it the best choice.



(a) Result after genetic search (b) Offset found by GSA (c) Final layout with the optimal NPV

Figure 4.25: *Resulting layouts after 1st and 2nd stage optimisation. Starting layout is the best individual found using GA.*

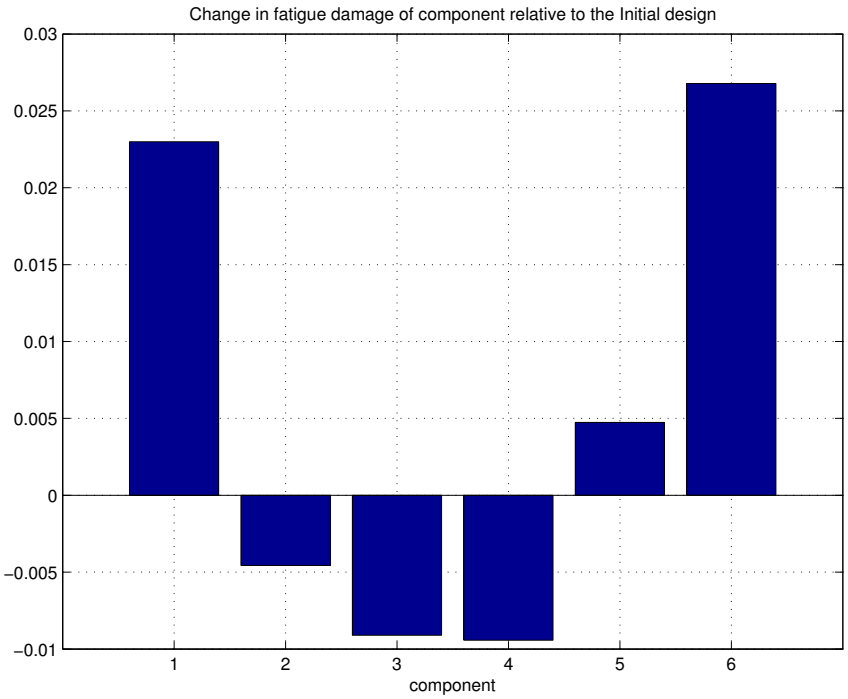


Figure 4.26: *Relative change in fatigue damage per component - optimal layout*

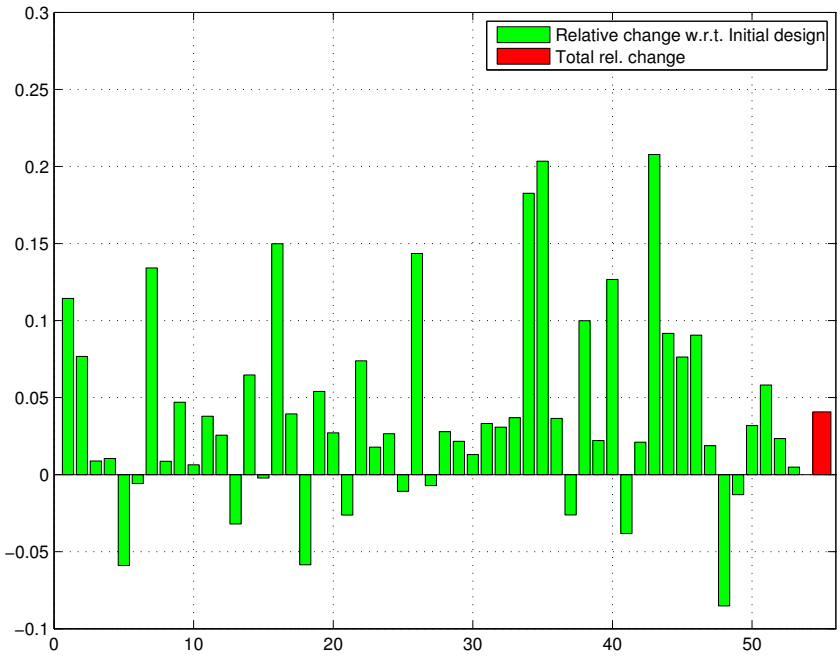
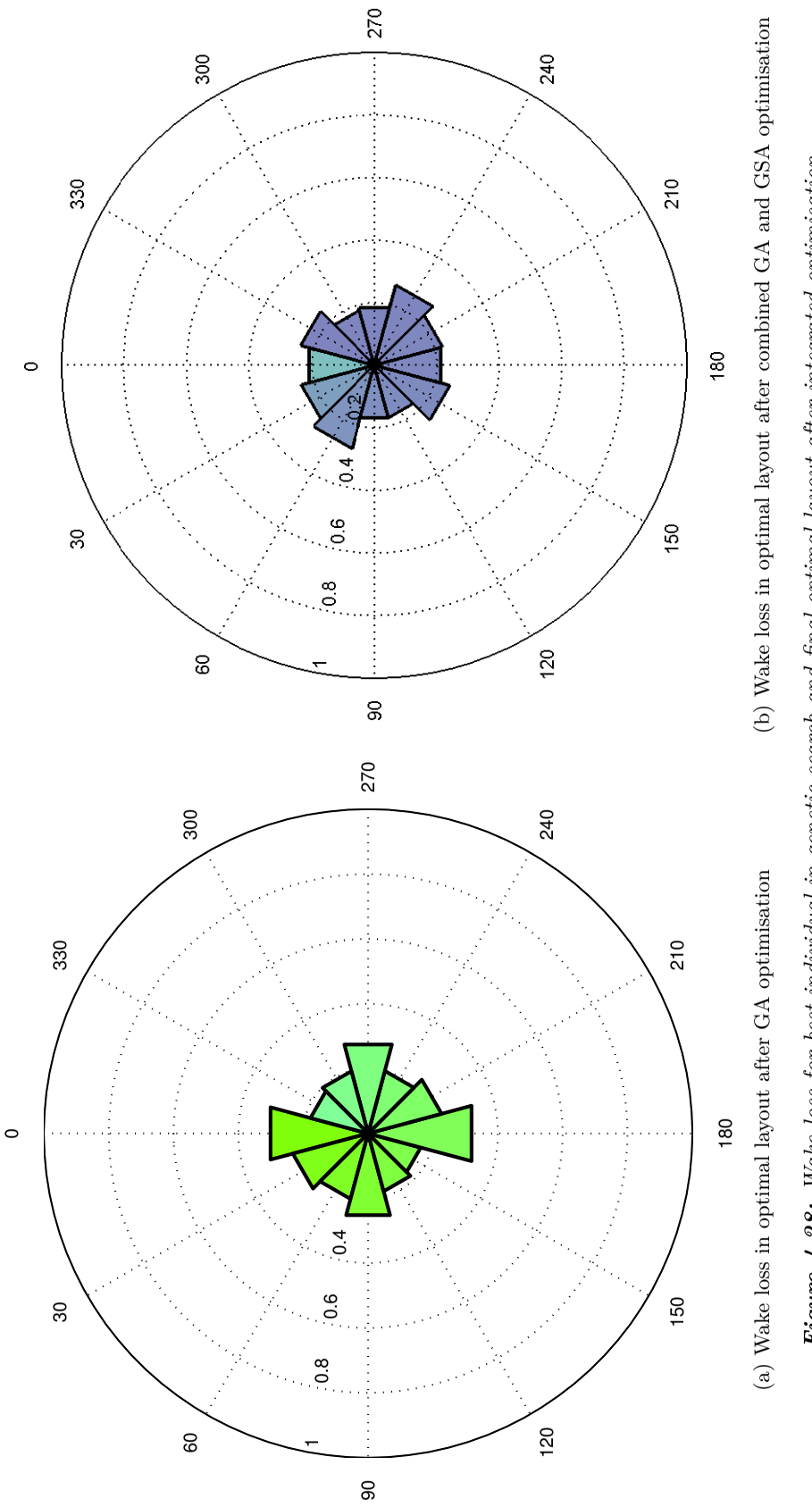


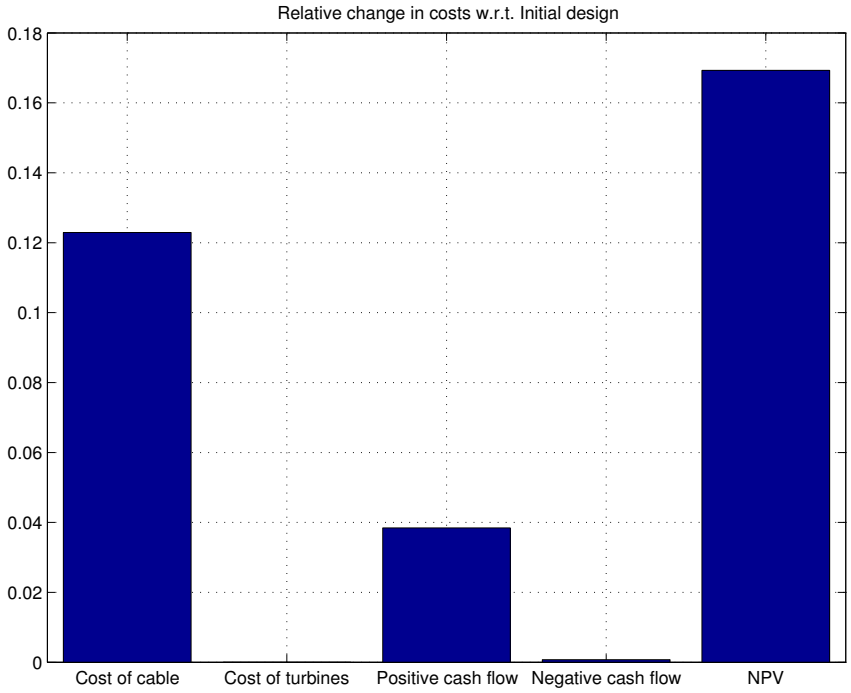
Figure 4.27: *Relative change in AEP per turbine - optimal layout*



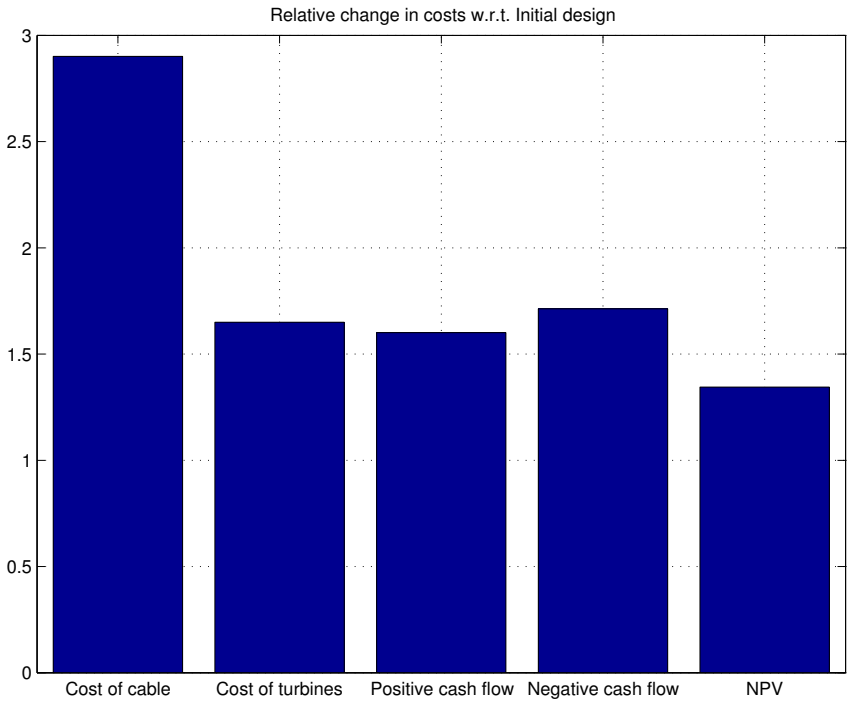
(a) Wake loss in optimal layout after GA optimisation

(b) Wake loss in optimal layout after combined GA and GSA optimisation

Figure 4.28: *Wake loss for best individual in genetic search and final optimal layout after integrated optimisation*



(a) GSA w.r.t. result from GA



(b) Total change w.r.t. scaled Middelgrunden layout

Figure 4.29: *Relative change in costs for optimal layout*

4.3.6 Summary and conclusion from layout optimisation

Two-stage optimisation platform is created with the aim of optimising wind farms both w.r.t. the number of turbines and their position within the allowed area. The objective of optimisation is to maximise the NPV. In the first stage, the attention is put on the number of turbines and finding the optimal balance between the profit and investment. For this purpose, a genetic search is implemented. In the second stage, the focus is primarily on minimising the wake effect, which is done by the gradient based algorithm.
The objective function is examined to sensitivity w.r.t. some characteristic wind farm related parameters, and the convexity of the function is confirmed by finding the deflection point after which the farm becomes saturated by the wake, and its economic feasibility starts to decrease.
The optimal layout, found as a result of combining these two strategies, is not characterised as the best layout w.r.t. parameters such as capacity factor or component degradation. Taking Middelgrunden as an example, the optimal wind farm would not result by saving the funds, but on the contrary, by almost tripling the investment. Naturally, there is a limit to how much the site can handle. This limit is heavily influenced by the weighting of costs for computing the NPV. The optimality will thus depend on how well the market trends can be predicted.
To summarise:

- The genetic search has a purpose of finding the optimal balance between costs and investment. As such, it depends highly on the weights used in computing the NPV.
- Gradient based optimisation does not influence the investment significantly and is reduced basically by minimising the wake effect which is done by breaking apart the clustered turbines.
- Combination of these two strategies, combined with the surrogate wake model, is a robust and fast solution for WFLO.

Chapter 5

Summary and Conclusions

A framework for optimisation of wind farm layouts has been built and presented. The building was done in two stages. In the first part, surrogate modeling was employed to create a DWM surrogate model as a way of reducing the computational demand required during the optimisation. In the second part, platform using multi-fidelity optimisation strategy was created by combining genetic algorithm for global search, and gradient based algorithm for local optimisation. Through the use of the genetic search, the layouts are optimised primarily to find the optimal number of turbines, but also to approximate the turbine position. Particular individuals from the search are then chosen, and further tuned through gradient based optimisation. The final result is the layout with the highest NPV. The optimisation framework as such was tested on a Middelgrunden wind farm. Allowing the turbine count to vary, an optimum is found by increasing the number of turbines and spreading them apart. The results are, however, very dependant of the weighting of the costs used to compute the objective function. Thus, for a real project, weights must be adjusted and costs carefully benchmark-ed.
The conclusions are summarised in the continuation.

- Planning has to be done considering financial parameters and market trends
- Apart from power production, component degradation throughout the lifetime and derived O&M costs must be taken into account
- As a consequence of required loading data, the wake model has to capture in-stationary flow field. As a consequence, Dynamic Wake Meandering model is used as a module in HAWC2 aeroelastic tool

- Due to the computational expenses of running wind turbine simulations in using HAWC2 and the DWM model, for optimisation purposes, a surrogate model is used as a replacement
- The surrogate model is created for seven outputs from HAWC2 simulations:
  - Equivalent load (Mx) at the blade root
  - Equivalent load (My) at the blade root
  - Equivalent load (Mx) at the tower bottom
  - Equivalent load (My) at the tower bottom
  - Equivalent load (Mx) at the shaft
  - Equivalent load (Mz) at the shaft
  - Power production
- Resulting model is cheap (1000 samples), fast to produce (modeling platform encompasses cluster computing), robust (interpolation scheme, analytic function), reasonably accurate (Mean Relative Error of 5%) and fast to evaluate (MATLAB function call, wind farm with over 80 turbines is analysed in under a minute)
- The optimisation has the objective is to maximise the Net Present Value of the wind farm. This is done considering investment (subject to the number of turbines and cable length), positive cash flows (determined from AEP which depends of turbine count and turbine positioning) and negative cash flows (originating from O&M costs which are derived from fatigue damage)
- Two-stage optimisation platform is set up. Genetic search finds the optimal number of turbines and optimal layout on a coarse discretized grid. Gradient based algorithm is used to further tune up the layout by shifting the turbines so the wake effect is minimised.
- First stage of optimisation is heavily influenced by weighting of the costs terms, however, the convexity of the function is confirmed
- During the genetic optimisation, grading of the individuals must be robust. First, because the NPV may have both positive and negative values. Second, because the population consists of individuals that vary to great extend in the beginning, and are equally optimal towards the end.

- To ensure genetic diversity, mapping of neighbouring turbines is implemented in the crossover, permutation is set as the mutation mechanism, and injection of permuted individuals is performed in case population becomes invariant.
- The bounds on the turbine count are slacken to study the optimality w.r.t. the turbine count and to ensure sufficient number of "healthy" individuals.
- The optimal layout of the cabling grid is determined by the Kruskal algorithm which uses the greedy approach of the minimum spanning tree
- Gradient based optimisation shifts the turbine (breaks the clusters) and minimises the wake effect. It has barely any effect to the investment, that is, by spacing the turbines further away it increases the cost of cabling.
- Due to non-linearity of the objective function, gradient estimation using finite difference approximation may not be the best solution. However, the surrogate modeling toolbox does not allow all surrogate types to be exported to analytic expression, but they remain a black-box function. Furthermore, because the cabling grid is determined by a separate function, application of algorithmic differentiations was not possible. In the future work on the topic, the cabling costs may be neglected, and dedicated surrogate modeling platform should be developed to result in a model given as an analytic function. This will result in a more precise computation of the gradients, and consequently in better solutions.

Bibliography

[1] IEC 61400-1 ed/3: Wind turbines - part 1: Design requirements.

[2] E. W. E. Association et al. *The economics of wind energy*. EWEA, 2009.

[3] J. W. Bandler, Q. S. Cheng, S. Dakroury, A. S. Mohamed, M. H. Bakr, K. Madsen, J. Søndergaard, et al. Space mapping: the state of the art. *Microwave Theory and Techniques, IEEE Transactions on*, 52(1):337–361, 2004.

[4] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to algorithms*. MIT Press, 2009.

[5] K. Crombecq. A gradient-based approach to adaptive surrogate modelling. Technical report, Technical report, University of Antwerp, 2008.

[6] K. Crombecq, L. De Tommasi, D. Gorissen, and T. Dhaene. A novel sequential design strategy for global surrogate modeling. In *Winter Simulation Conference*, pages 731–742. Winter Simulation Conference, 2009.

[7] A. I. Forrester and A. J. Keane. Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences*, 45(1):50–79, 2009.

[8] A. A. Giunta, S. F. Wojtkiewicz, M. S. Eldred, et al. Overview of modern design of experiments methods for computational simulations. In *Proceedings of the 41st AIAA aerospace sciences meeting and exhibit, AIAA-2003-0649*, 2003.

[9] D. Goldberg. *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley Publ, 1989.

[10] D. Gorissen, I. Couckuyt, P. Demeester, T. Dhaene, and K. Crombecq. A surrogate modeling and adaptive sampling toolbox for computer based design. *The Journal of Machine Learning Research*, 11:2051–2055, 2010.

[11] D. Gorissen, I. Couckuyt, E. Laermans, and T. Dhaene. Multiobjective global surrogate modeling, dealing with the 5-percent problem. *Engineering with Computers*, 26(1):81–98, 2010.

[12] R. L. Graham and F. F. Yao. Finding the convex hull of a simple polygon. *Journal of Algorithms*, 4(4):324–331, 1983.

[13] A. S. Hedayat, N. J. A. Sloane, and J. Stufken. *Orthogonal arrays: theory and applications*. Springer Science & Business Media, 1999.

[14] V. Jarník. O jistém problému minimálním. *Práca Moravské Prírodovedecké Spolecnosti*, 6:57–63, 1930.

[15] J. M. Jonkman, S. Butterfield, W. Musial, and G. Scott. *Definition of a 5-MW reference wind turbine for offshore system development*. National Renewable Energy Laboratory Golden, CO, 2009.

[16] Q. Y. Kenny, W. Li, and A. Sudjianto. Algorithmic construction of optimal symmetric latin hypercube designs. *Journal of statistical planning and inference*, 90(1):145–159, 2000.

[17] S. Koziel and L. Leifsson. Surrogate-based modeling and optimization. *Applications in Engineering*, 2013.

[18] S. Koziel and X.-S. Yang. *Computational optimization, methods and algorithms*, volume 356. Springer Science & Business Media, 2011.

[19] J. B. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical society*, 7(1):48–50, 1956.

[20] G. Larsen. A simple generic wind farm cost model tailored for wind farm optimization, 2009.

[21] G. C. Larsen, H. Aagaard Madsen, F. Bingöl, and W. E. D. R. D. Risø National Lab., DTU. *Dynamic wake meandering modeling*. 2007.

[22] G. C. Larsen, H. Aagaard Madsen, N. Troldborg, T. J. Larsen, P.-E. Réthoré, P. Fuglsang, S. Ott, J. Mann, T. Buhl, M. Nielsen, H. Markou, J. N. Sørensen, K. S. Hansen, R. F. Mikkelsen, V. Okulov, W. Z. Shen, M. Heath, J. King, G. McCann, W. Schlez, I. Carlén, H. Ganander, E. Migoya, A. Crespo, A. Jiménez, J. Prieto, A. Stidworthy, D. Carruthers, J. Hunt, S. Gray, D. Veldkamp, A. S. Mouritzen, L. Jensen, T. Krogh, B. Schmidt, K. Argyriadis, and P. Frohböse. Topfarm - next generation design tool for optimisation of wind farm topology and operation, 2011.

[23] G. C. Larsen and P.-E. Réthoré. Topfarm – a tool for wind farm optimization. *Energy Procedia*, 35:317–324, 2013.

[24] H. Larsen and L. Sønderberg Petersen. *DTU International Energy Report 2014: Wind energy — drivers and barriers for higher shares of wind in the global power generation mix*. Technical University of Denmark, 2014.

[25] T. J. Larsen and A. M. Hansen. *How 2 HAWC2, the user's manual*. Risø National Laboratory, 2007.

[26] S. Lundberg. Performance comparison of wind park configurations. Technical report, Chalmers University of Technology, 2003.

[27] J. Mann. Spatial structure of neutral atmospheric surface-layer turbulence. *Journal of Fluid Mechanics*, 273:141–168, 1994.

[28] A. Matlab, K. Toolbox, S. N. Lophaven, H. B. Nielsen, and J. Sndergaard. Dace-a matlab kriging toolbox. 2002.

[29] M. L. Minsky and S. A. Papert. *Perceptrons - Expanded Edition: An Introduction to Computational Geometry*. MIT press Boston, MA:, 1987.

[30] P. Morthorst. The economics of wind power. *Proceedings (on Cd-rom)*, 2003.

[31] H. M. Nguyen, I. Couckuyt, L. Knockaert, T. Dhaene, D. Gorissen, and Y. Saeys. An alternative approach to avoid overfitting for surrogate models. In *Proceedings of the Winter Simulation Conference*, pages 2765–2776. Winter Simulation Conference, 2011.

[32] A. B. Owen. Orthogonal arrays for computer experiments, integration and visualization. *Statistica Sinica*, 2(2):439–452, 1992.

[33] A. F. Pease. The science of prediction. http://www.siemens.com/innovation/pool/en/publikationen/publications_pof/pof_fall_2011/machine_learning/pof0211_ml_prognosen_en.pdf, 2011.

[34] N. V. Queipo, R. T. Haftka, W. Shyy, T. Goel, R. Vaidyanathan, and P. K. Tucker. Surrogate-based analysis and optimization. *Progress in aerospace sciences*, 41(1):1–28, 2005.

[35] C. E. Rasmussen. Gaussian processes for machine learning. 2006.

[36] P.-E. Rethore, P. Fuglsang, G. C. Larsen, T. Buhl, T. J. Larsen, and H. A. Madsen. Topfarm: Multi-fidelity optimization of wind farms. *WIND ENERGY*, 17(12):1797–1816, 2014.

[37] C. Rüth. A better forecast for renewable energy generation. http://www.siemens.com/innovation/en/home/pictures-of-the-future/energy-and-efficiency/sustainable-power-generation-neural-networks.html, 2014.

[38] A. Saltelli, M. Ratto, T. Andres, F. Campolongo, J. Cariboni, D. Gatelli, M. Saisana, and S. Tarantola. *Introduction to Sensitivity Analysis*, pages 1–51. John Wiley & Sons, Ltd, 2008.

[39] M. Samorani. The wind farm layout optimization problem. In *Handbook of Wind Power Systems*, pages 21–38. Springer, 2013.

[40] T. W. Simpson, J. Poplinski, P. N. Koch, and J. K. Allen. Metamodels for computer-based engineering design: survey and recommendations. *Engineering with computers*, 17(2):129–150, 2001.

[41] V. N. Vapnik. An overview of statistical learning theory. *Neural Networks, IEEE Transactions on*, 10(5):988–999, 1999.

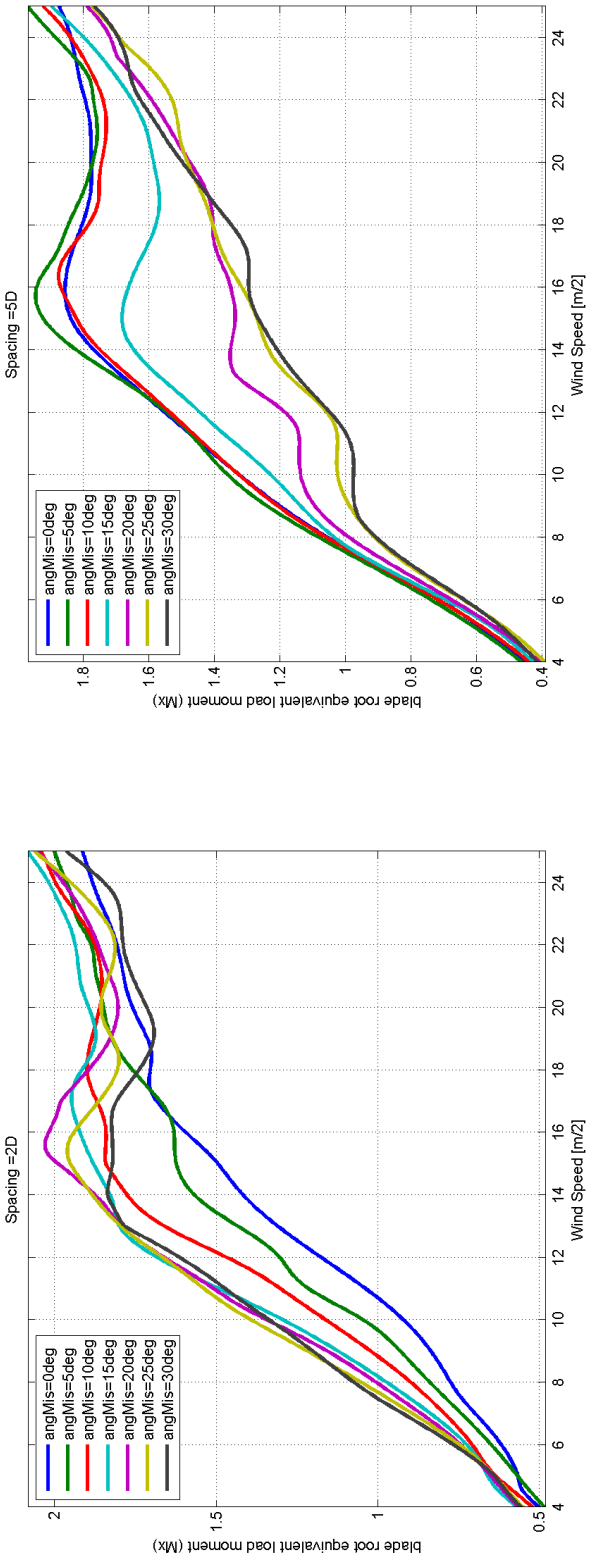
[42] B. Vocking, H. Alt, M. Dietzfelbinger, C. Reischuk, Scheideler, H. Vollmer, and D. Wagner. Algorithms unplugged. *Algorithms Unplugged, Algorith. Unplugged*, pages 1–406, 2011.

[43] S. M. Wild, R. G. Regis, and C. A. Shoemaker. Orbit: Optimization by radial basis function interpolation in trust-regions. *SIAM Journal on Scientific Computing*, 30(6):3197–3219, 2008.

Appendix A

Surrogate model plots

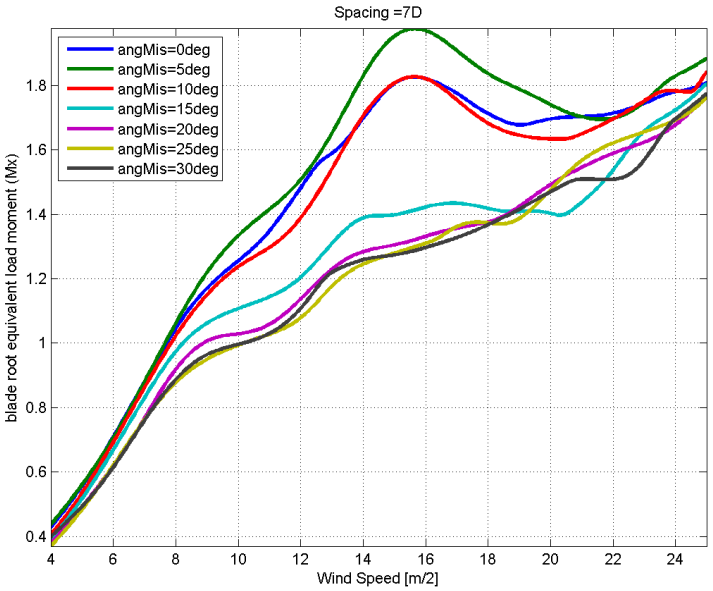
Plots of each output are given in Figure A.1 to Figure A.21. Note that all results are normalized with the conditions a turbine experiences at free ambient wind speed of 10m/s (no wake).



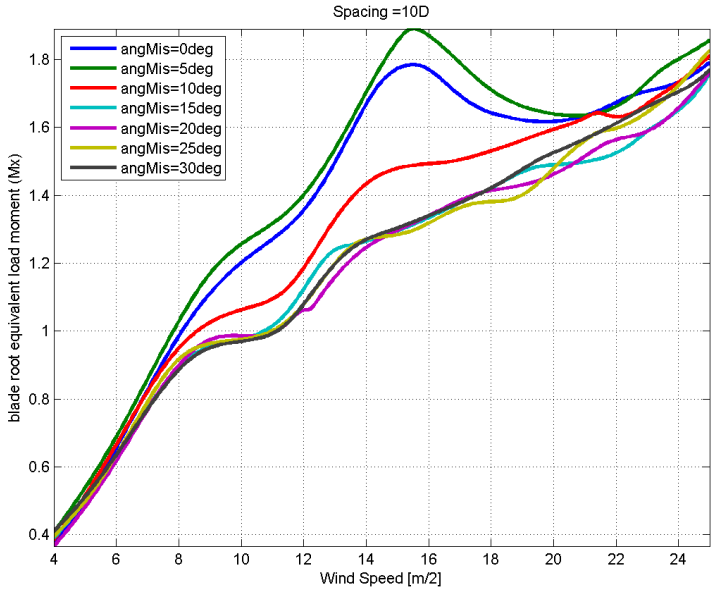
(a) 2D spacing

(b) 5D spacing

Figure A.1: *Lifetime equivalent moment at the blade root (Mx)*

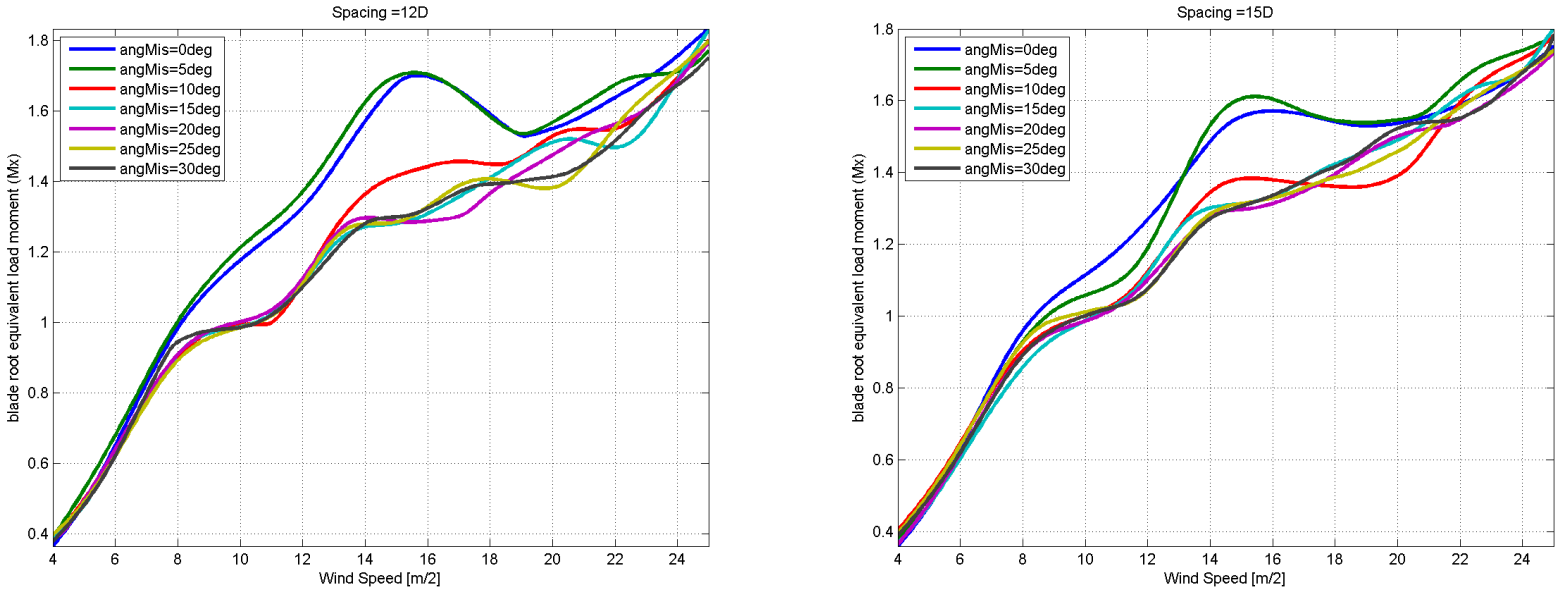


(a) 7D spacing



(b) 10D spacing

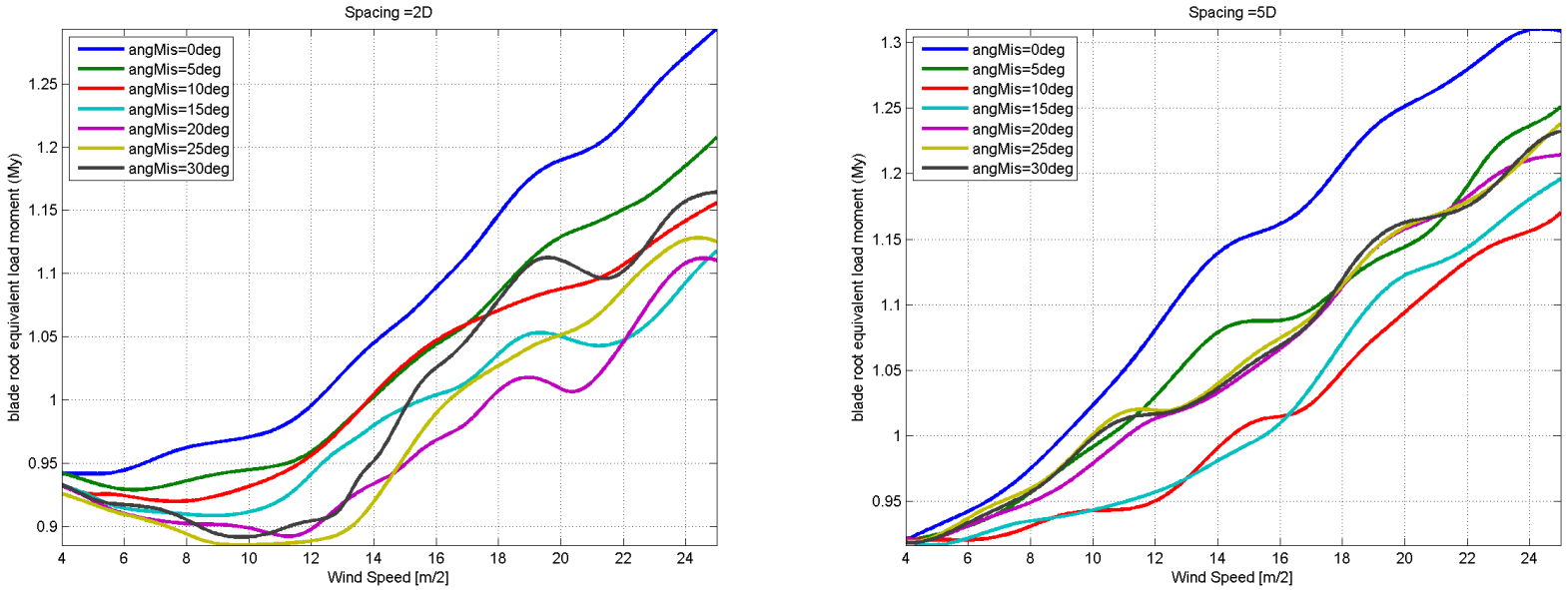
Figure A.2: *Lifetime equivalent moment at the blade root (Mx)*



(a) 12D spacing

(b) 15D spacing

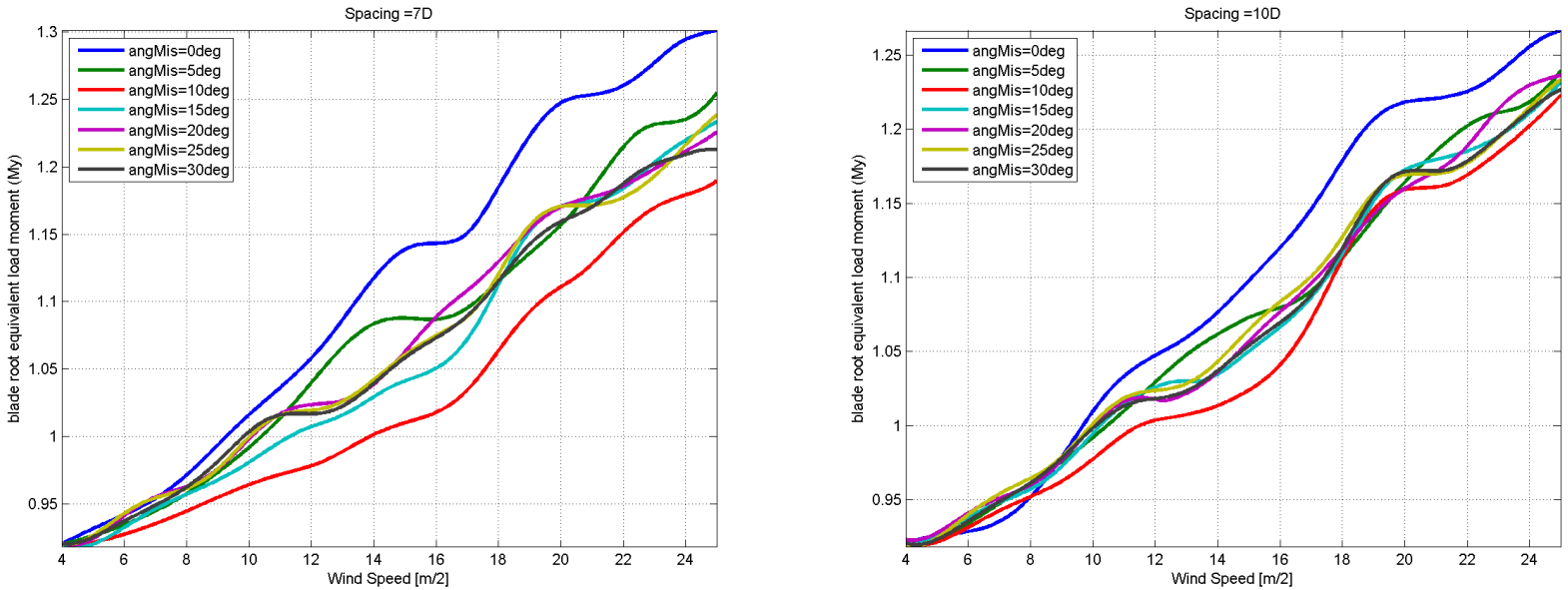
Figure A.3: *Lifetime equivalent moment at the blade root (Mx)*



(a) 2D spacing

(b) 5D spacing

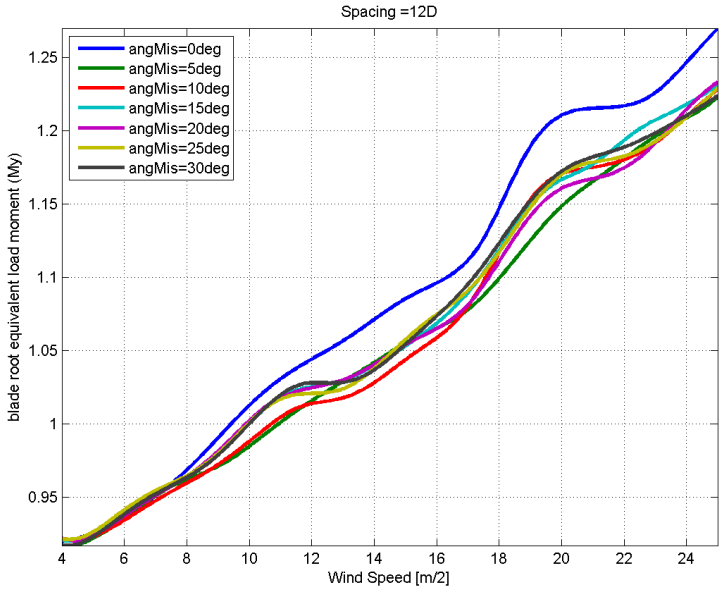
Figure A.4: *Lifetime equivalent moment at the blade root (My)*



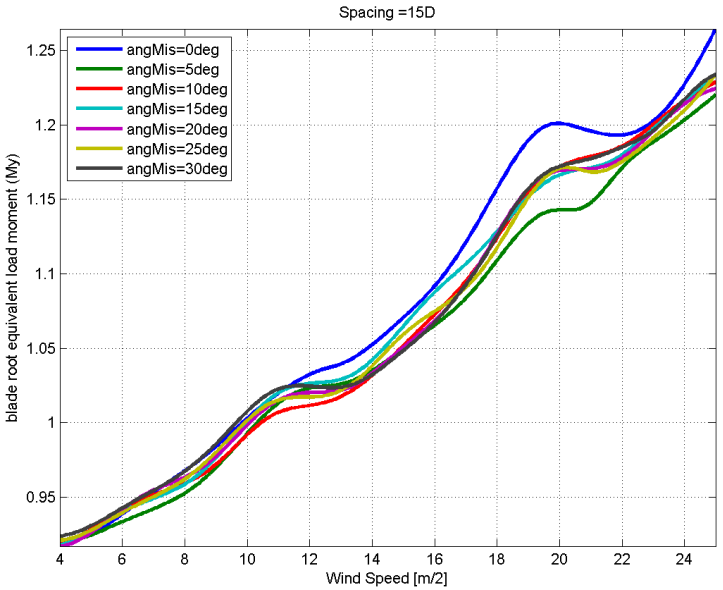
(a) 7D spacing

(b) 10D spacing

Figure A.5: *Lifetime equivalent moment at the blade root (My)*

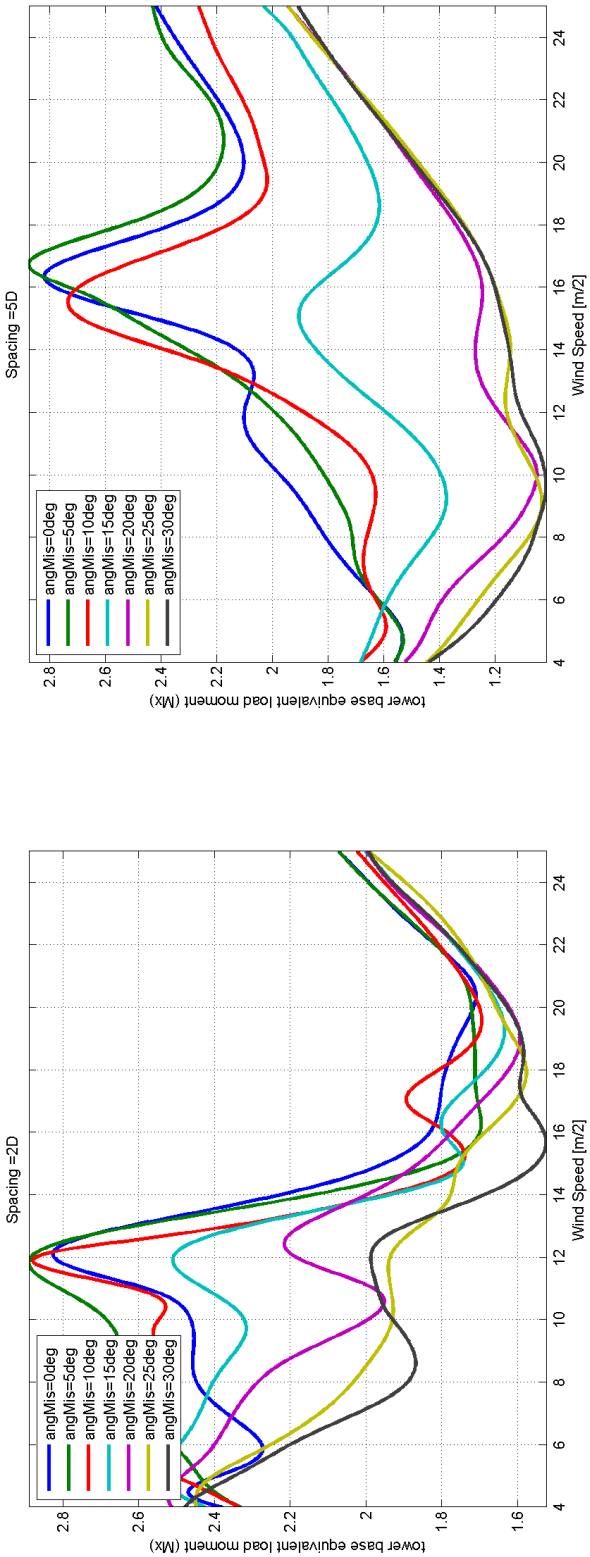


(a) 12D spacing



(b) 15D spacing

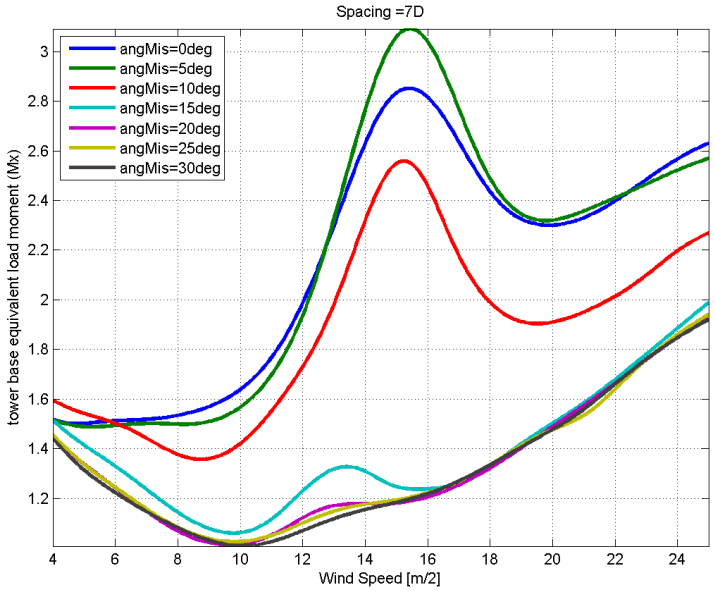
Figure A.6: *Lifetime equivalent moment at the blade root (My)*



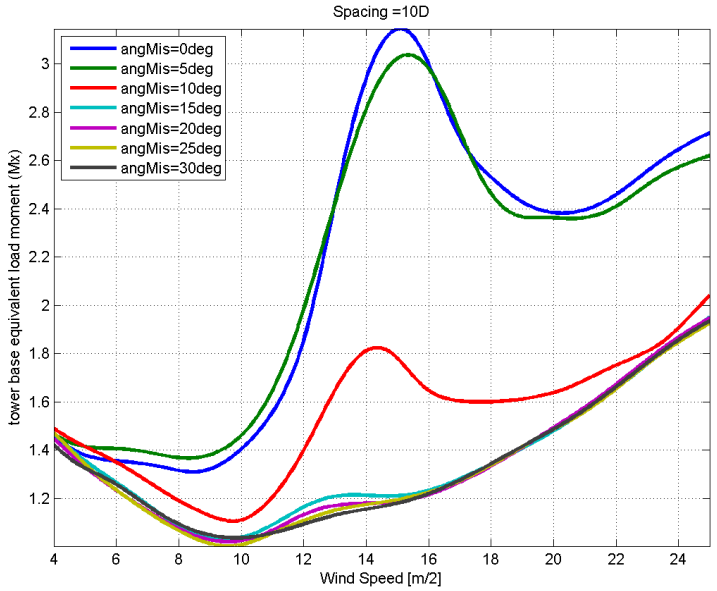
(a) 2D spacing

(b) 5D spacing

Figure A.7: *Lifetime equivalent moment at the blade root (Mx)*

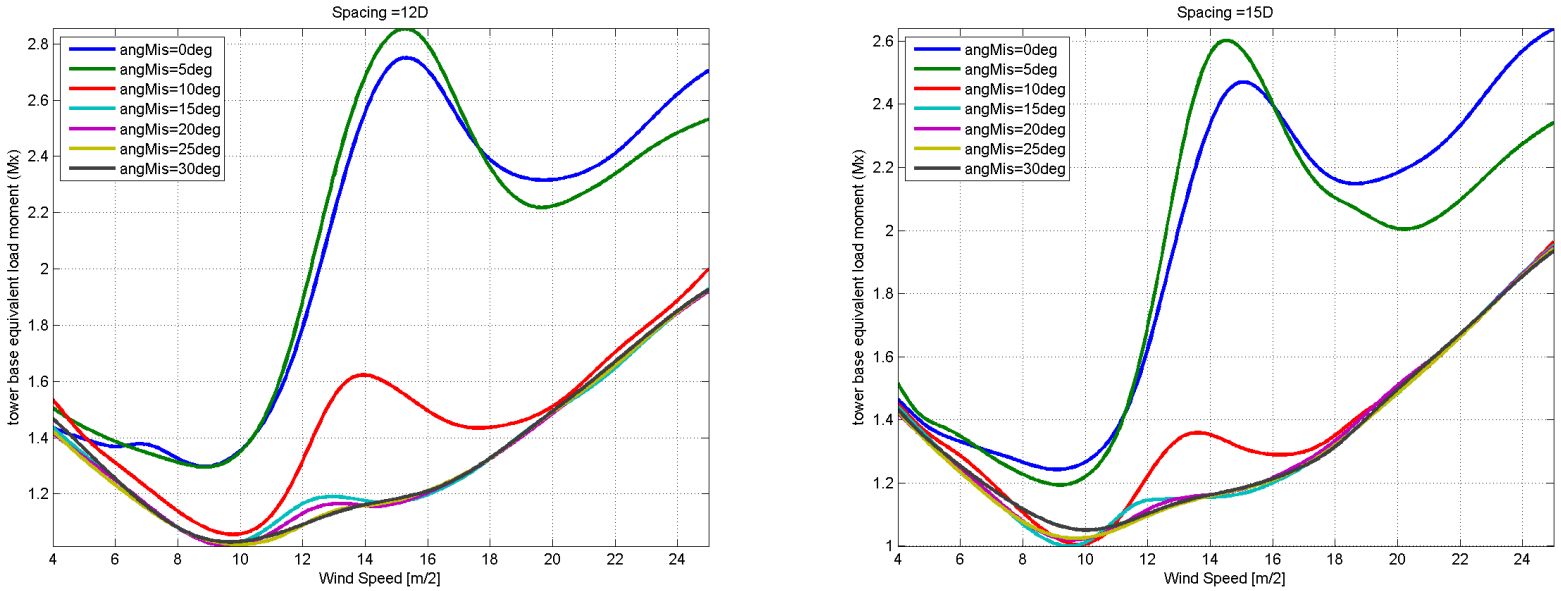


(a) 7D spacing



(b) 10D spacing

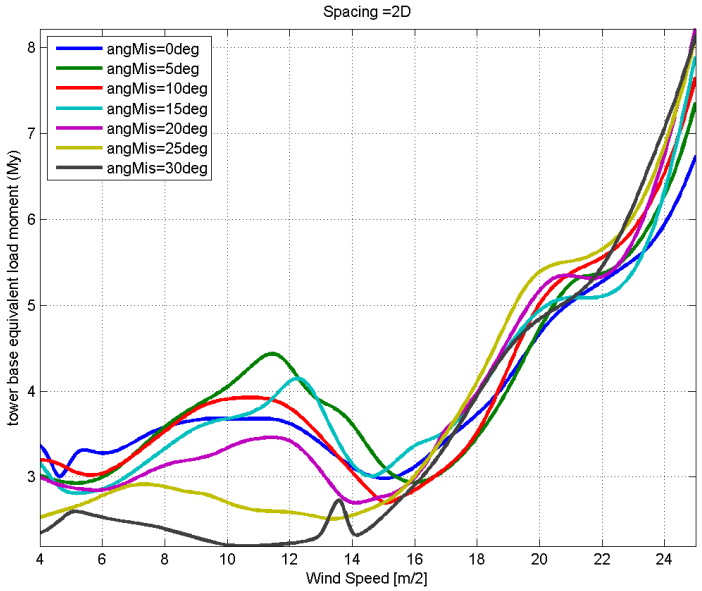
Figure A.8: *Lifetime equivalent moment at the blade root (Mx)*



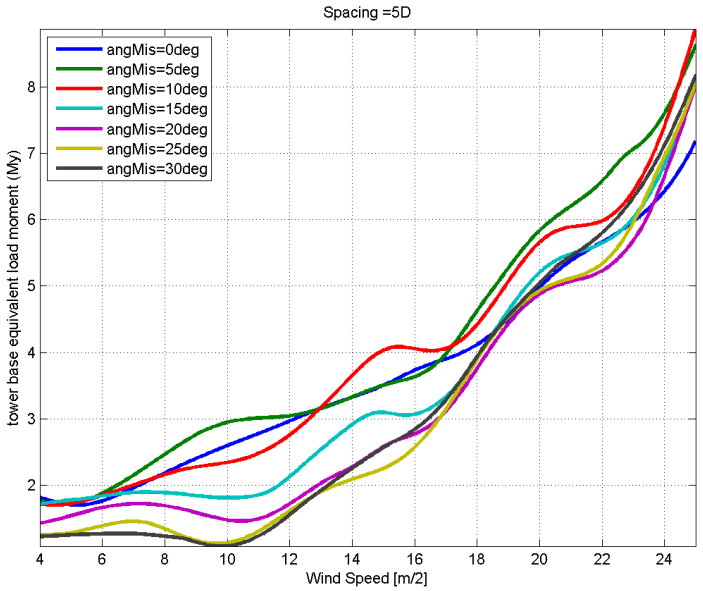
(a) 12D spacing

(b) 15D spacing

Figure A.9: *Lifetime equivalent moment at the blade root (Mx)*

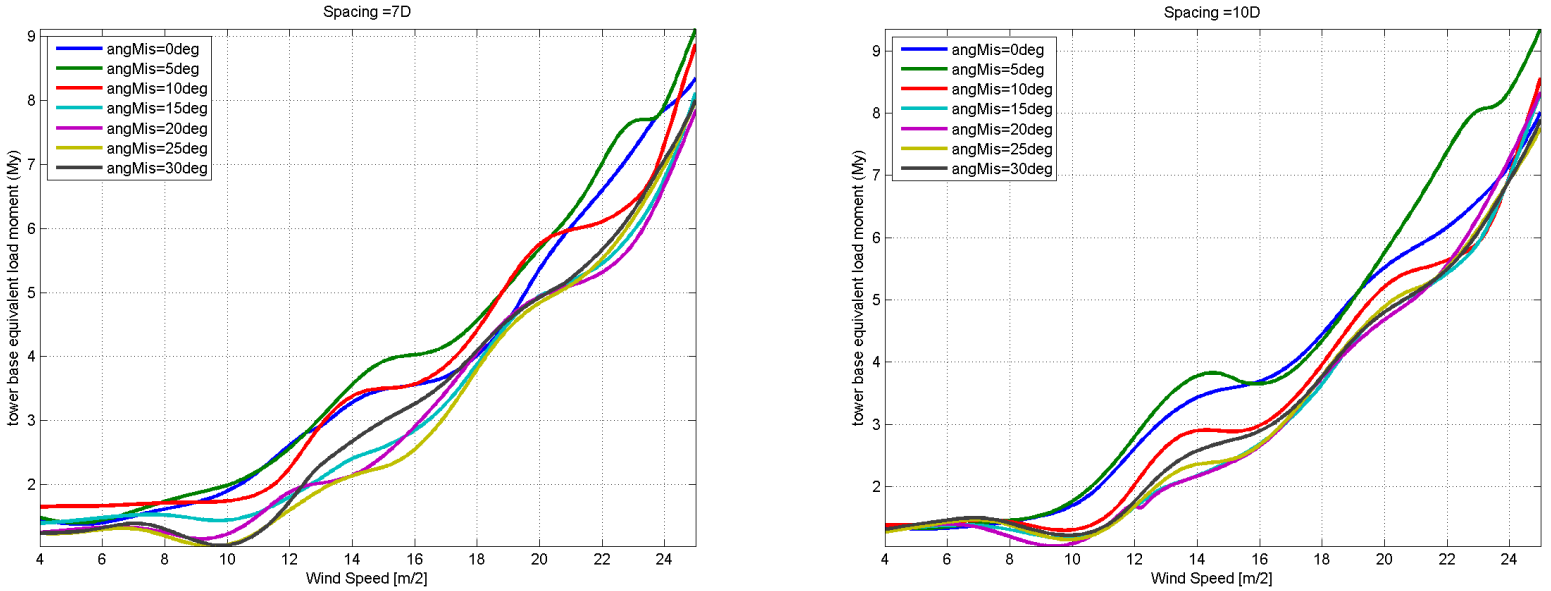


(a) 2D spacing



(b) 5D spacing

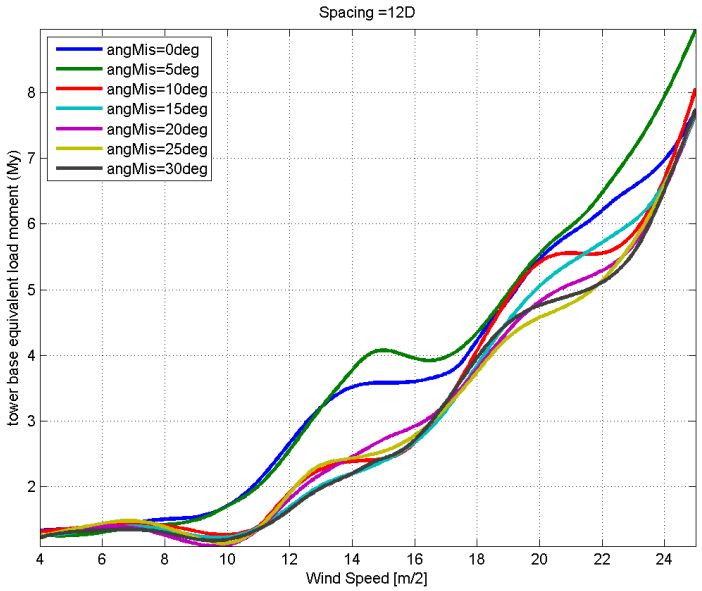
Figure A.10: *Lifetime equivalent moment at the blade root (My)*



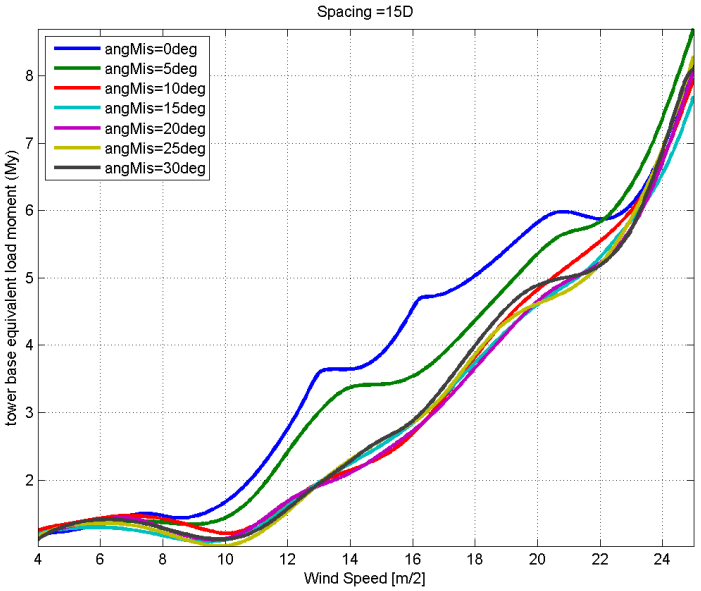
(a) 7D spacing

(b) 10D spacing

Figure A.11: *Lifetime equivalent moment at the blade root (My)*

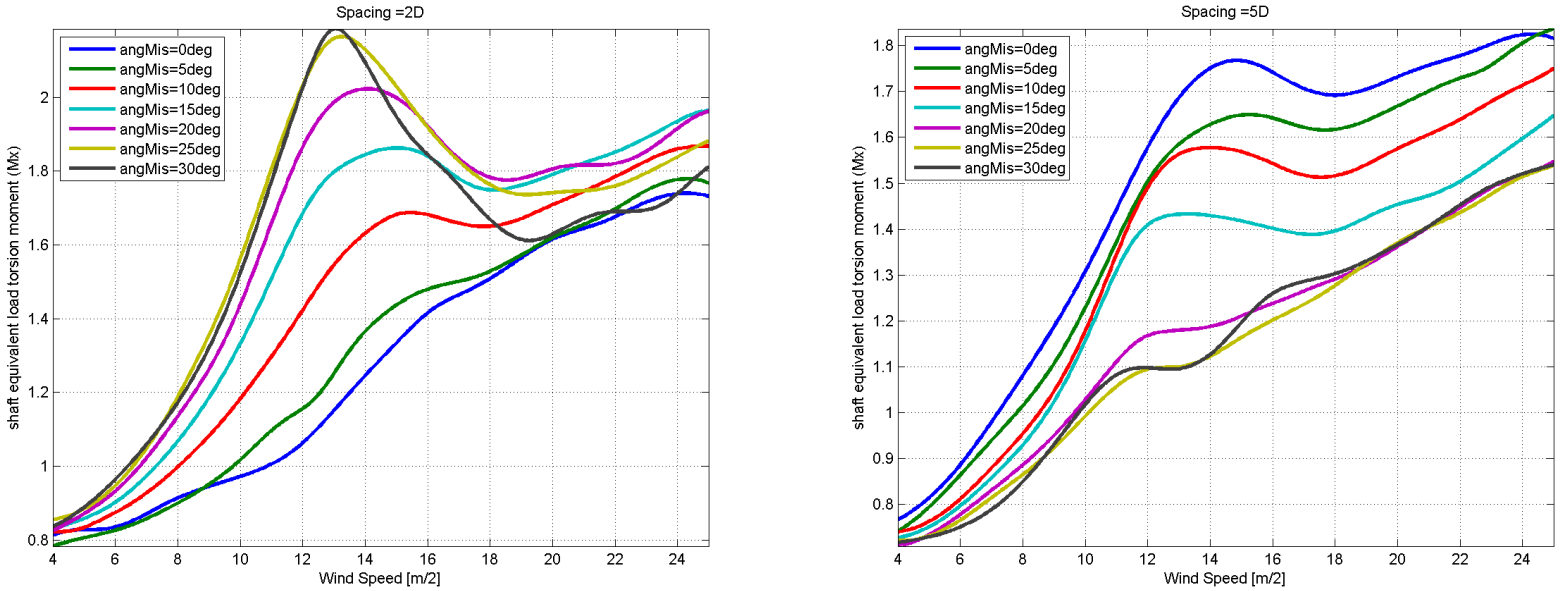


(a) 12D spacing



(b) 15D spacing

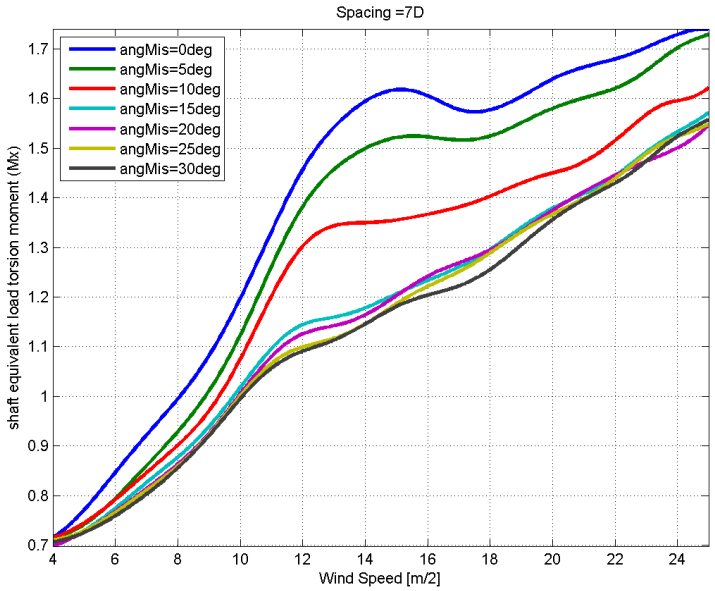
Figure A.12: *Lifetime equivalent moment at the blade root (My)*



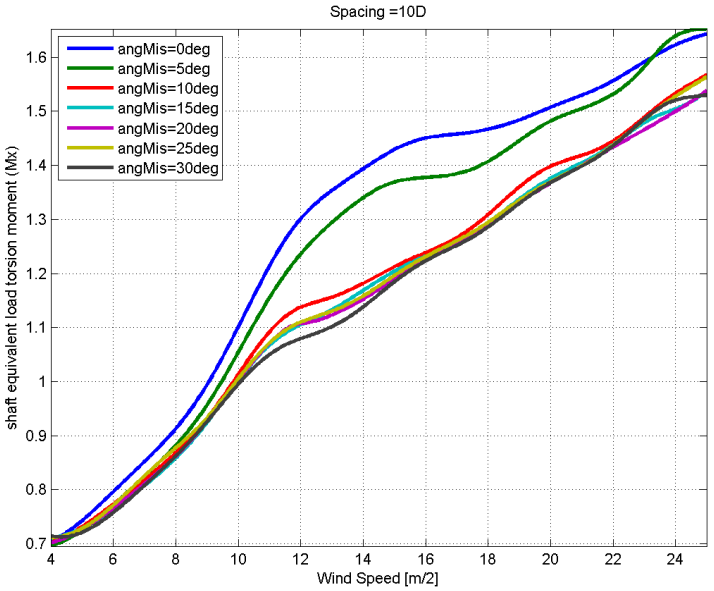
(a) 2D spacing

(b) 5D spacing

Figure A.13: *Lifetime equivalent moment at the blade root (Mx)*

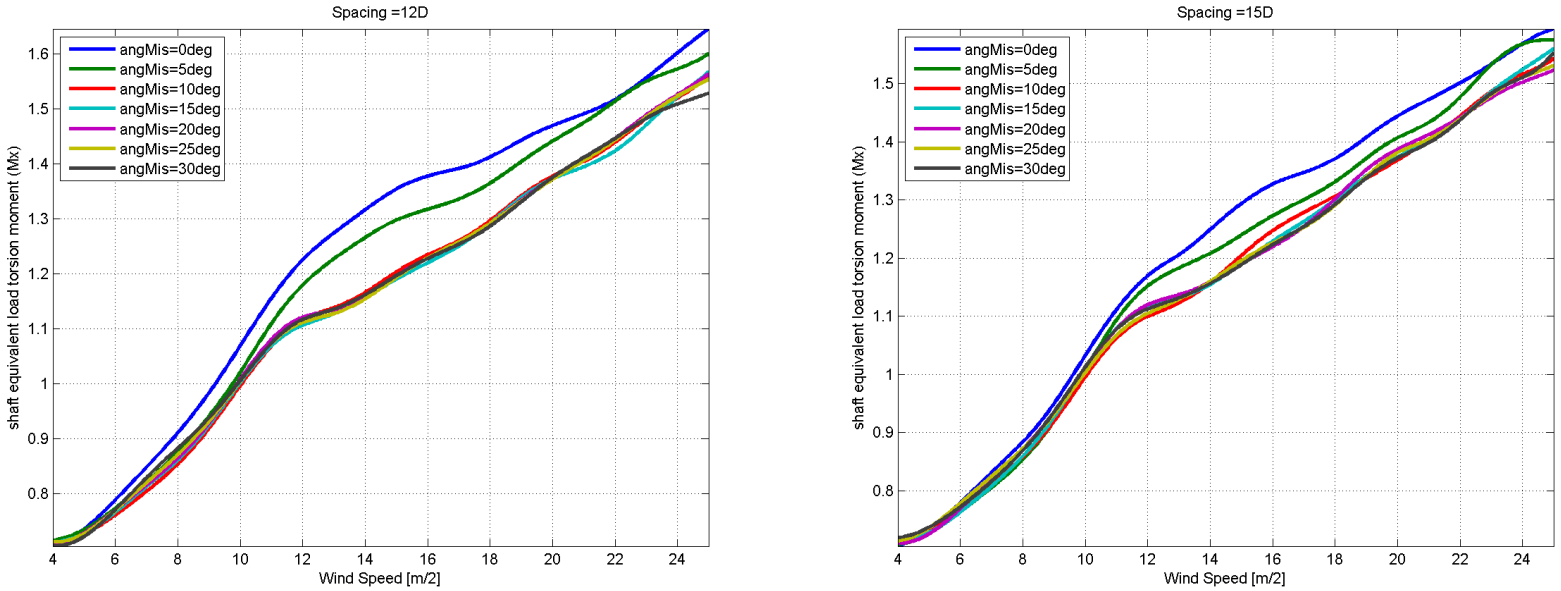


(a) 7D spacing



(b) 10D spacing

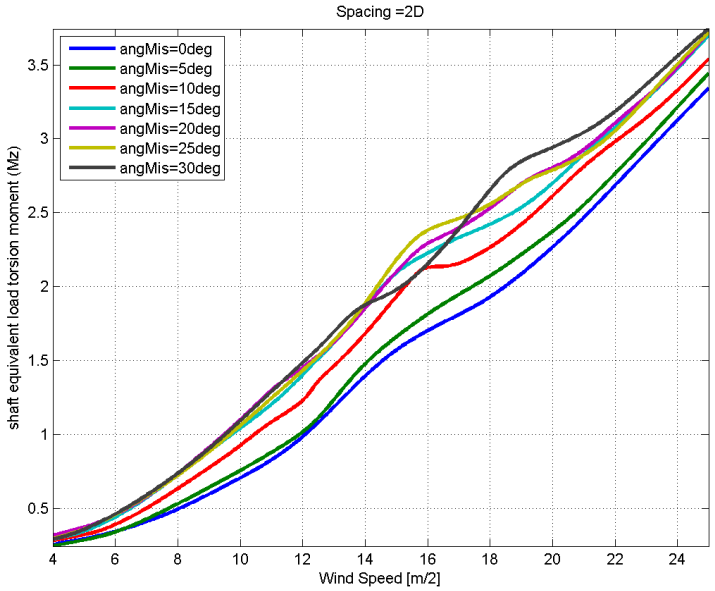
Figure A.14: *Lifetime equivalent moment at the blade root (Mx)*



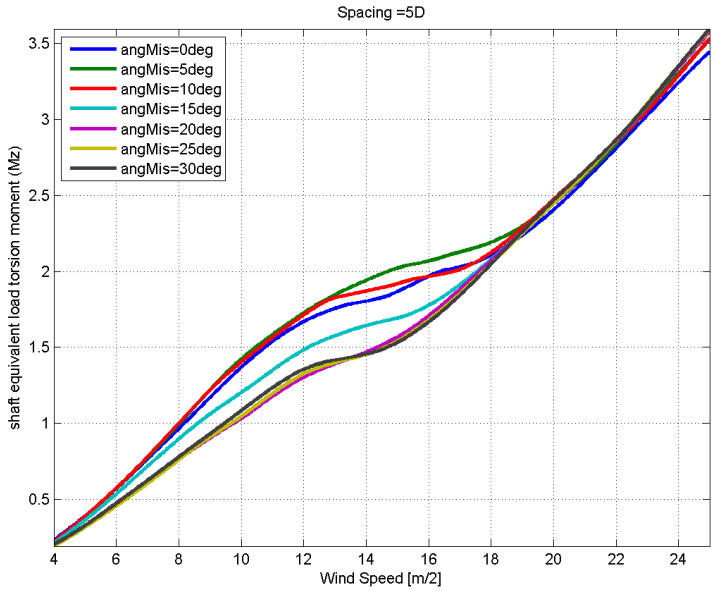
(a) 12D spacing

(b) 15D spacing

Figure A.15: *Lifetime equivalent moment at the blade root (Mx)*

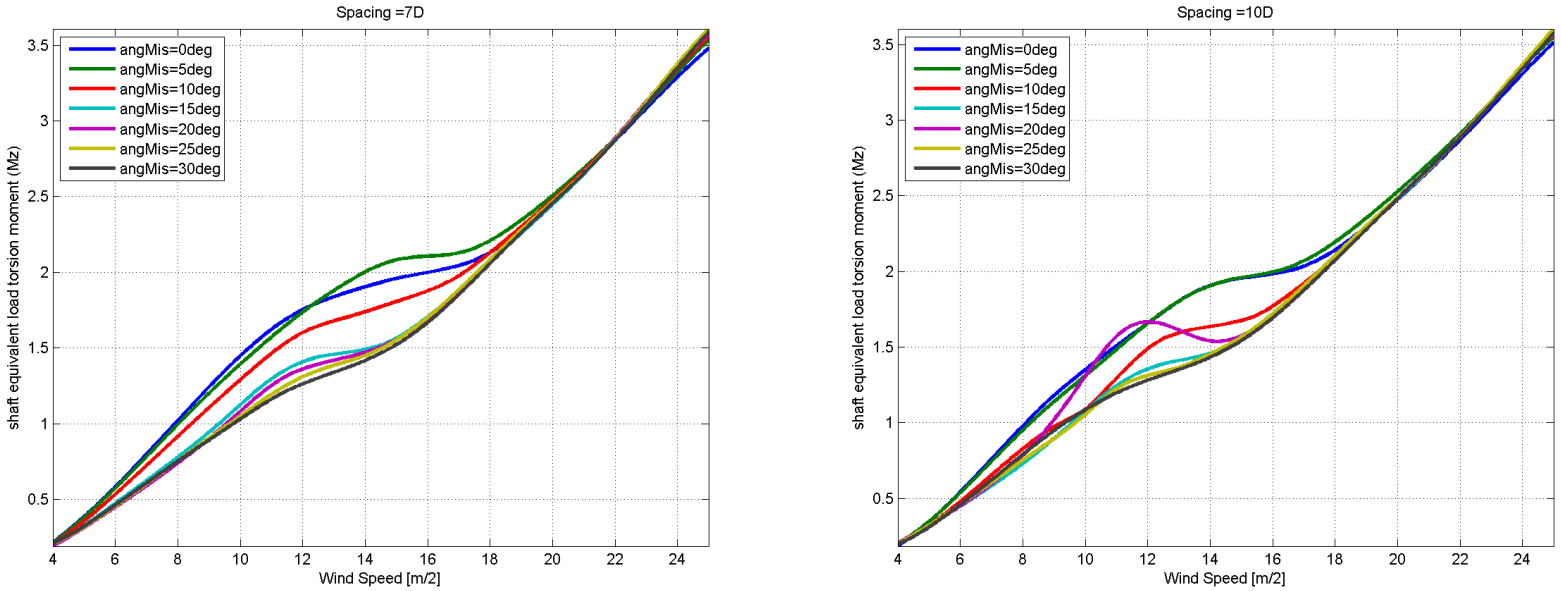


(a) 2D spacing



(b) 5D spacing

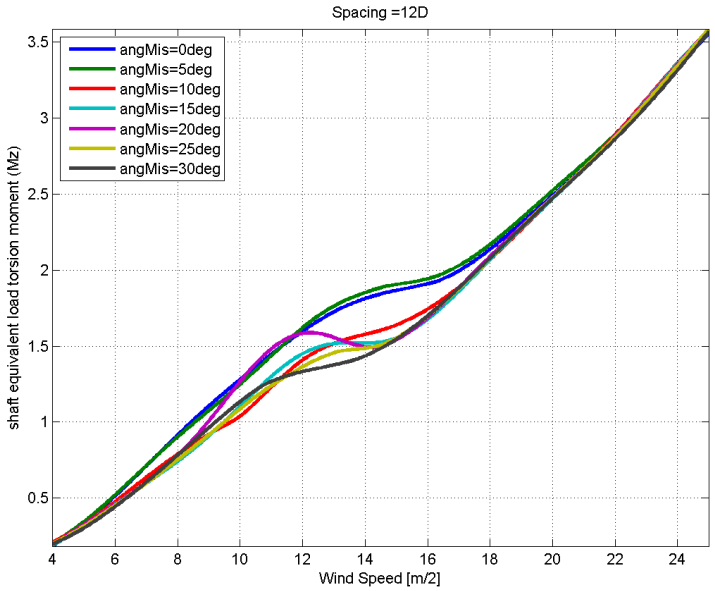
Figure A.16: *Lifetime equivalent moment at the blade root (Mz)*



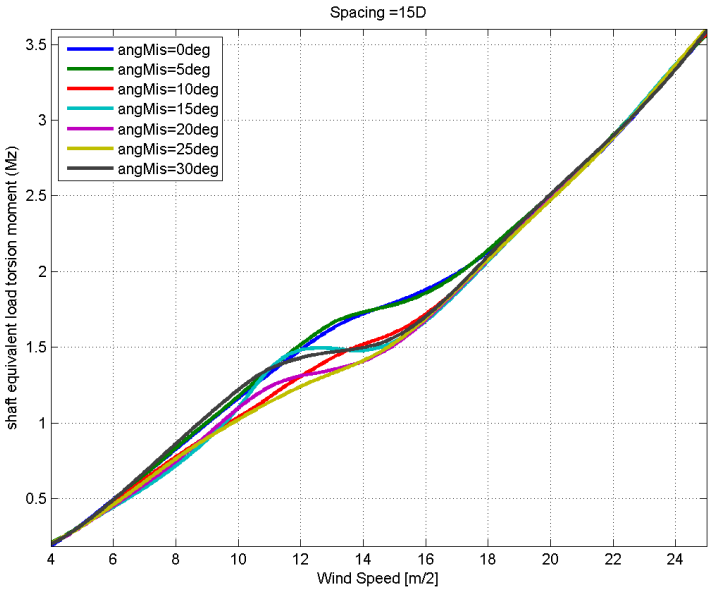
(a) 7D spacing

(b) 10D spacing

Figure A.17: *Lifetime equivalent moment at the blade root (Mz)*

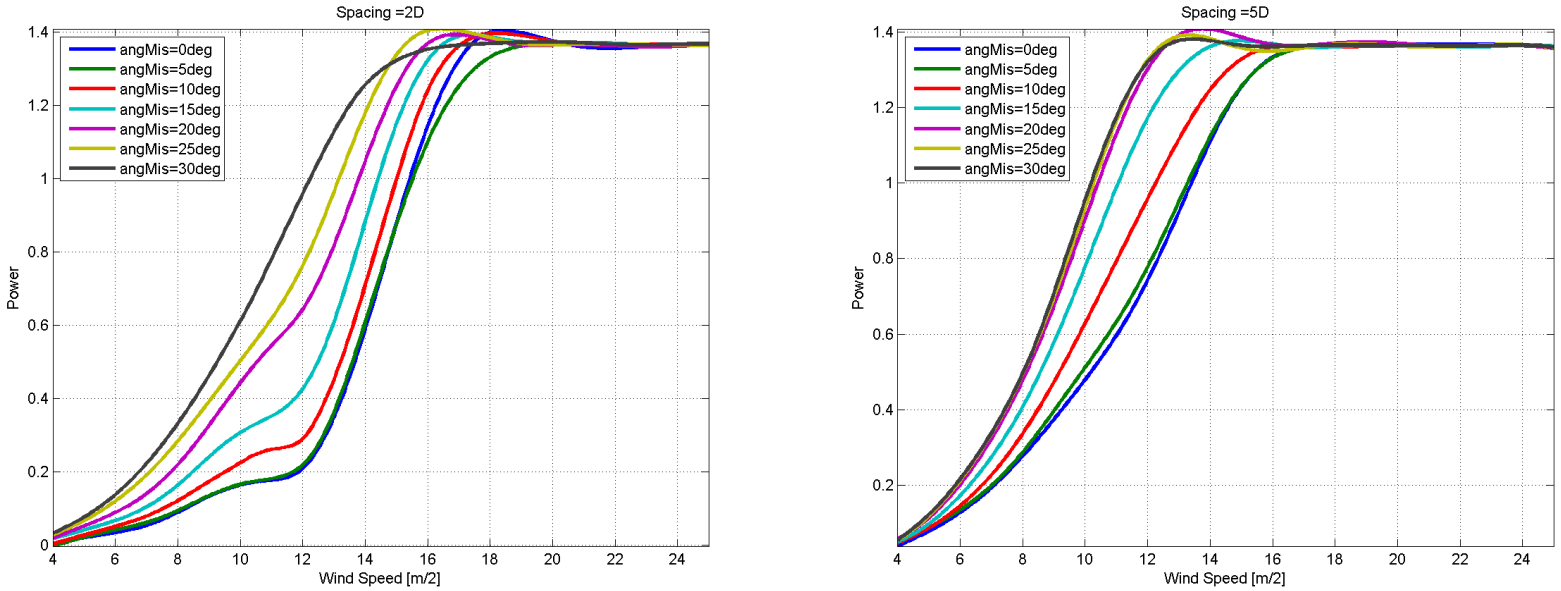


(a) 12D spacing



(b) 15D spacing

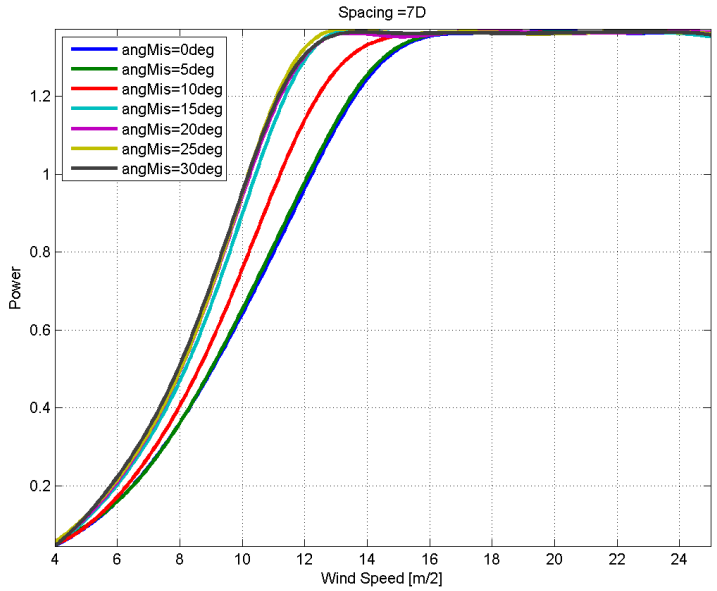
Figure A.18: *Lifetime equivalent moment at the blade root (Mz)*



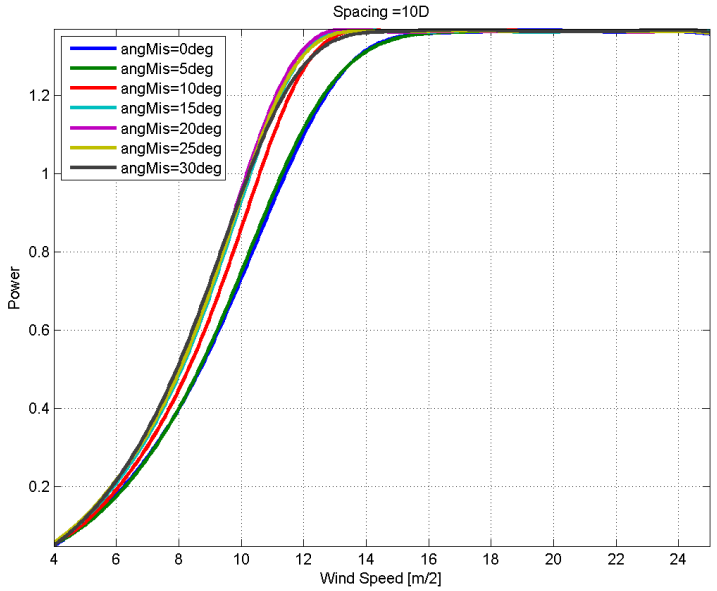
(a) 2D spacing

(b) 5D spacing

Figure A.19: *Power output vs. ambient wind speed [m/s]*

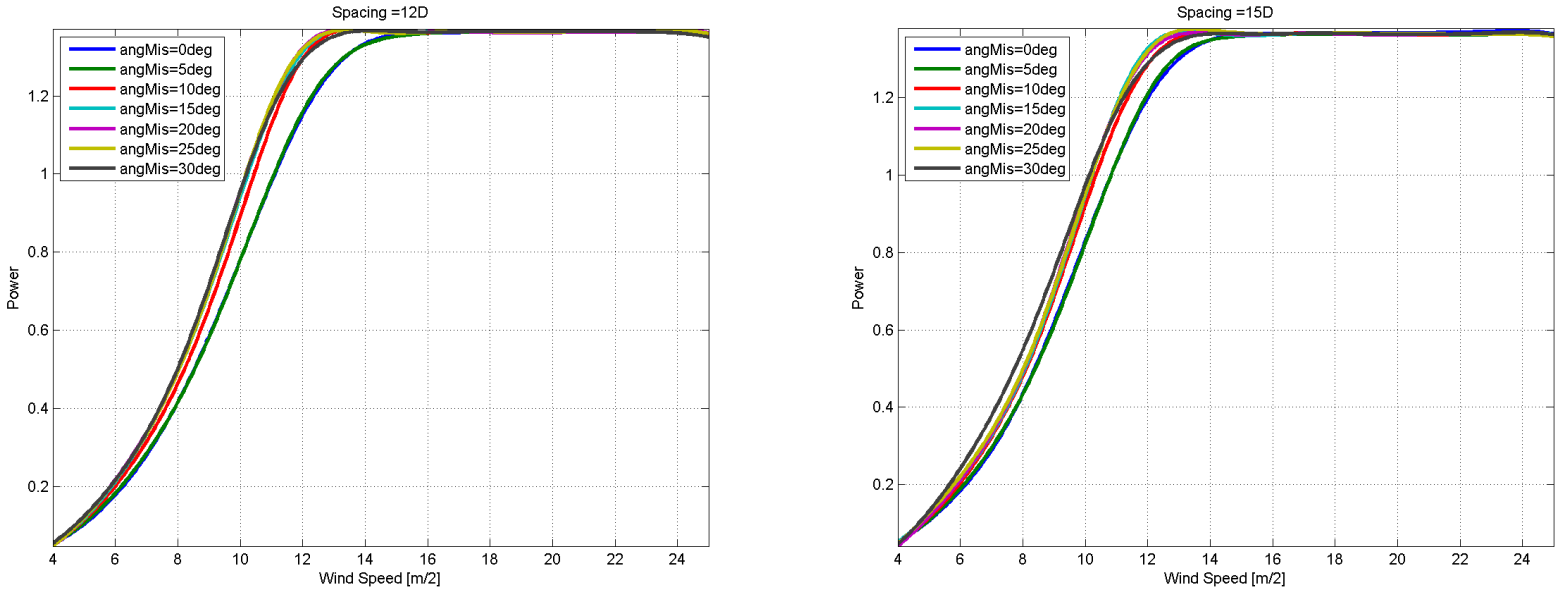


(a) 7D spacing



(b) 10D spacing

Figure A.20: *Power output vs. ambient wind speed [m/s]*



(a) 12D spacing

(b) 15D spacing

Figure A.21: *Power output vs. ambient wind speed [m/s]*

Appendix B

Reports

B.1 Surrogate model report - example

Supplied as a separate document. Printed in continuation

B.2 WFLO report - example

Supplied as a separate document. Printed in continuation

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

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

Danmarks Tekniske Universitet
DTU Vindenergi
Nils Koppels Allé
Bygning 403
2800 Kgs. Lyngby
Telefon 45 25 25 25

info@vindenergi.dtu.dk
www.vindenergi.dtu.dk