ISSN 0280-5316
ISRN LUTFD2/TFRT--5883--SE

Distributed Control of Wind Farm

Benjamin Biegel

Department of Automatic Control
Lund University
June 2011

Lund University
Department of Automatic Control
Box 118
SE-221 00 Lund Sweden

Document name
MASTER THESIS

Date of issue
June 2011

Document Number
ISRN LUTFD2/TFRT--5883--SE

Author(s)
Benjamin Biegel

Supervisor
Jakob Stoustrup Aalborg University, Denmark. Daria Madjidian Automatic Control Lund, Sweden. Anders Rantzer Automatic Control Lund, Sweden (Examiner)

Sponsoring organization

Title and subtitle
Distributed Control of Wind Farm (Distribuerad reglering av vindkraftspark)

Abstract
The growing use of wind energy as major energy source, has led to the construction of large wind farms. The control of such wind farms is typically separated into control at wind turbine level and at wind farm level. The wind farm controller assures that the wind farm produces the demanded amount of power, by providing set-points for the turbines in the farm. The local turbine control assures that the local set-point is tracked. Current wind farm controllers conduct the power distribution in a static manner. In this work it is explored, how fatigue reduction in the wind farm can be achieved, by letting the controller distribute the power demand dynamically. In the first part of this work, the problem of reducing the fatigue on the turbines by dynamic power distribution is presented. A controller is designed based on this problem formulation, taking both tower and shaft fatigue into account. The controller is designed, such that it is both modular and scalable. This has the advantage of the controller being identical on all turbines in the wind farm. Also, the modularity allows turbines to be added or removed from the wind farm, without changing the controllers on the remaining turbines. Evaluation of the controller in a realistic simulation environment verifies the functionality of the controller, and shows that fatigue reductions of the tower and shaft in the magnitude of 10 % and 50 % respectively, can be expected. The final part of this work examines how the concept of fatigue reduction through dynamic wind farm control can be implemented on the offshore wind farm Thanet. Simulations indicate that the limitations of the server system at Thanet does not render this type of control possible.

Keywords

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280-5316

ISBN

Language
English

Number of pages
79

Recipient's note

Security classification

Preface

This thesis is the result of my 10th semester at Aalborg University. The project has been conducted as a study abroad at Lund University in Lund, Sweden.

The work consists of this thesis along with an appended CD.

I owe my supervisor, prof. Jakob Stoustrup great thanks for his support through the course of the semester, and especially for setting up the connection to Lund University. I also give great thanks to all the people at Lund University, who have made the stay at Lund University possible. Anders Rantzer and Daria Madjidian for many hours of support through the project, and for including me in the Aeolus experiments. Eva Westin for helping with all practical aspects and for finding an apartment in Lund.

Also I want to thank Joachim Løvgaard, for designing the beautiful cover pages.

Finally, I want to thank *Siemens A/S Fond* and *Otto Mønsteds Fond*, for supporting my studies financially.

Contents

Chapter 1

Introduction

The earth's fossil energy resources are limited and large resistance exists towards the use of nuclear power plants, supported by recent events. This calls for alternative ways of meeting the world's increasing need of power. Renewable energy is expected to play a big role in solving this problem [Sti08], [Bul01].

With an exponential growth over a long period of years, wind energy is a promising source of renewable energy [Sti08]. A large amount of research in the area of wind turbine control has therefore been conducted. The typical focus of this research is power production maximization, power quality optimization, and fatigue minimization, see *e.g.* [BSJB01], [WWB10], [NCVT10], [XXZ+08], [JPP08], [BJS+11].

For economic reasons, it is desirable to place wind turbines close to each other. This has led to the construction of wind farms, where large numbers of wind turbines operate together to meet some total power demand. The control of such wind farms is the topic of this thesis.

The wind farm control must assure, that the wind turbines in the farm produce the demanded power. This is accomplished by distributing the total power demand among the turbines in the wind farm, by providing each turbine with a power set-point. The turbines in the farm then use their local wind turbine controller to track this power set-point.

When the wind farm power demand is less than the available power, the controller is free to distribute the power set-points among the turbines, as long as the farm power demand is met. This introduces a freedom in the wind farm control. In current wind farms this freedom is not exploited, as the power distribution is done in a static manner based on long term measurements and predictions [KBS09], [HSBF05]. The focus of this thesis is to benefit from this freedom, by designing a wind farm controller that reduces the fatigue on the turbines in the wind farm, while meeting the farm power demand.

The first part of the thesis is based on a modular control strategy developed in [MMR11]. The control strategy achieves wind farm fatigue minimization by dynamically changing the power production set-points of the turbines in the wind farm. This is done in a distributed manner, based on communication between neighboring turbines in the wind farm. In this work, this method is described, modified and evaluated through realistic wind farm simulations.

In the second part, the focus is the development of a controller suitable for real wind farm experiments. These experiments are planned to be performed at Thanet wind farm in the UK in the summer of 2011. As the wind turbines at Thanet are not constructed with dynamic control in mind, the turbines only offer a limited possibility to apply this type of control, which must be taken into account in the controller design. Based on these limitations, a controller is designed and evaluated through simulations.

Chapter 2

Overview

In this chapter, the concepts of wind farm control and distributed control are introduced. Further, this chapter describes the scope of the two parts of this work. The first part concerns modular distributed controller design, while the subject of the second part is a distributed controller designed for Thanet wind farm.

2.1 Wind Farm Control

Wind Farm Control Overview

In a wind farm, a large number of wind turbines are positioned close to each other. The control of such a wind farm is illustrated in Figure 2.1. The turbines in the wind farm are illustrated to the right, each producing the power P^i_{out}, thus producing a total power of $P_{\text{farm,out}} = \sum_{i=1}^{N} P^i_{\text{out}}$, where N is the number of turbines. Also, each turbine experiences some measure of fatigue J^i, which means that the total fatigue in the farm becomes $J_{\text{farm}} = \sum_{i=1}^{N} J^i$. The fatigue can describe the mechanical loads on the turbines, the quality of the power, etc.

The control of the wind farm is split up into farm level control and turbine level control. The wind farm control distributes power set-points to all the turbines in the farm, *i.e.* turbine i is given the set-point P^i_{dem} etc. The local turbine controller in each wind turbine ensures that the output power of turbine P^i_{out} tracks the given power set-point P^i_{dem}. At the same time, the local wind turbine controller seeks other objectives, *e.g.* ensures structural stability.

As illustrated by 2.1, the wind farm controller distributes the power set-points P^i_{dem} based on the power demand to the whole wind farm $P_{\text{farm,dem}}$. Also various measurements from the turbines in the farm $y_{\text{meas},i}$, and often also from a meteorological mast, are available to the wind farm controller.

Currently the wind farm controller determines the power set-points P^i_{dem} based on wind measurements in the wind farm and on long term wind predictions. Based on these and the demand to the whole wind farm $P_{\text{farm,dem}}$, the wind farm controller provides static power set-points to all the turbines in the farm.

In this work we examine how to exploit the freedom in choosing the power set-points dynamically, such that the fatigue on the turbines in the wind farm is reduced, while honoring the power demand of the network operator.

By the terminology of Figure 2.1, we can roughly state that the current wind farm controllers ensure that $P_{\text{farm,out}} = P_{\text{farm,dem}}$ by static power distribution. In this work, we seek to use

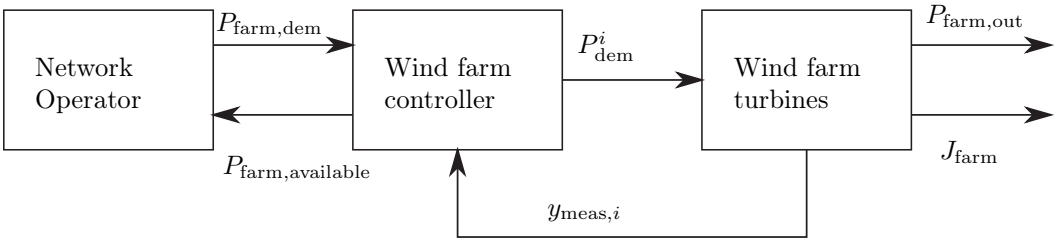

Figure 2.1: Illustration of a wind farm controller, operating based on measurements and on the power demanded by the network operator. The figure is based on [KBS09].

feedback to control the power set-points P^i_{dem} dynamically, leading to a minimization of J_{farm}, while still ensuring that $P_{\text{farm,out}} = P_{\text{farm,dem}}$.

Wind Farm Control Illustration

To illustrate the concept of a wind turbine control, the following simple fictive example is considered.

A wind farm consisting of five wind turbines is controlled by a wind farm controller. This is currently very roughly done in the following manner. Based on measurements from each wind turbine, the total available power in the farm can be determined. If there is more power available than demanded, each turbine is asked to generate a fraction of the power available to the given turbine.

Figure 2.2 illustrates this concept, where the five wind turbines are demanded to produce a total of 10 MW. For each bar, the height illustrates the power available for each given turbine, while the blue part is the power set-point given to the turbine. The sum of the power set-points of the five turbines, *i.e.* the blue parts, then correspond to the power demand. The power set-points are plotted over time in Figure 2.3 (left).

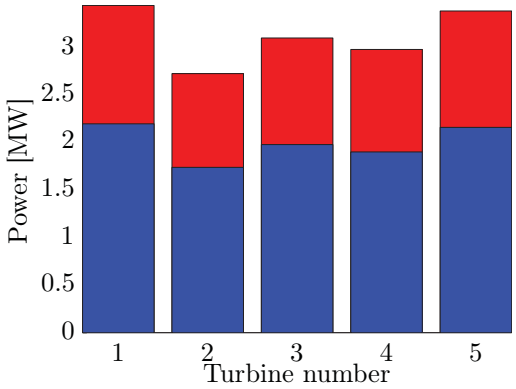

Figure 2.2: Illustration of a wind farm consisting of five wind turbines, each with a given amount of wind power available, represented by the total height of the bars. To meet the demand of 10 MW, all turbines are asked to produce a given fraction of the power available at each turbine (blue).

In this work, we allow the power demands to the turbines to vary over time. For this example with five wind turbines, the power demands could look like the illustration in Figure 2.3 (right). Still, the turbines produce the demanded 10 MW, but the freedom in varying the power set-points

is used to minimize the fatigue on the turbines.

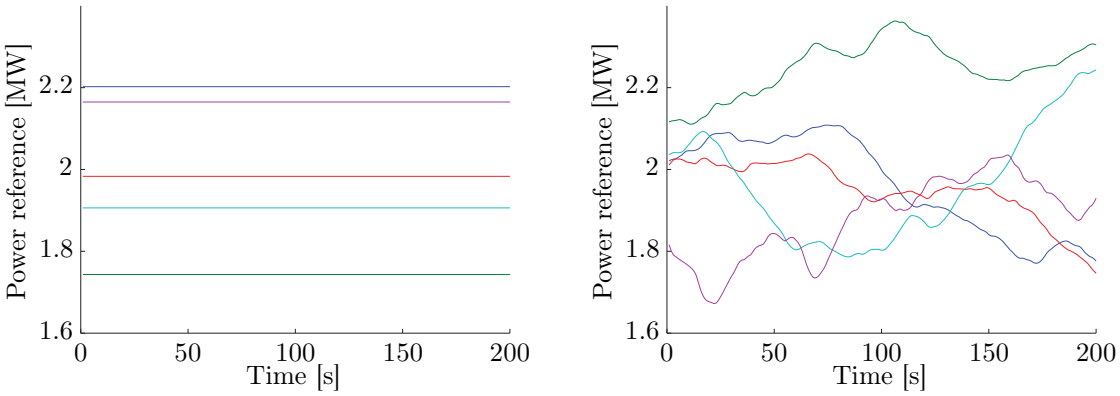

Figure 2.3: The power set-points to the five turbines in the farm, in the static case (left) and the dynamic case (right). In both places, the power output must equal the total power demand.

2.2 Distributed Control

Typically, control theory can be classified as either centralized or decentralized. In the centralized case, the controller has access to all measurements and decides all actuator inputs. In the decentralized case, we consider a number of dynamic units that are somehow coupled. Each unit is able to access some set of the measurements and control some set of actuators based on this [Swi10].

In a wind farm, each turbine can be seen as a unit in a distributed system. The coupling of these units is caused by the shared wind field and by the power demand to the wind farm. Further it is assumed that each wind turbine has access to local measurements and measurements from a number of neighboring wind turbines. Hereby each wind turbine has access to some set of the farm measurements. Based on the information available at each wind turbine, the distributed wind farm controller is able to dynamically update the power set-point on the individual turbines.

We illustrate this concept of centralized and distributed control, again by looking at five wind turbines controlled by a dynamic wind farm controller.

To dynamically determine the power set-points for the turbines in the farm P^i_{dem}, the wind farm controller must have access to wind farm measurements. Also, it must be able to distribute the power set-point to the wind turbines in the farm.

In centralized control, all turbines will send their measurements to the central controller, which will determine the power set-points P^i_{dem} and return these to the individual turbines in the farm. This centralized design is illustrated in Figure 2.4 (left).

In distributed control, each turbine contains a part of the controller. Based on this controller and based on local measurements and measurements from some given number of neighboring turbines, each turbine locally determines the power set-point. This concept is illustrated for the case of five wind turbines in Figure 2.4 (right). In the example presented in this figure, the turbines are allowed to communicate with their neighboring turbines.

By choosing a distributed wind farm approach, it is possible to construct a control structure that is both modular and scalable [MMR11]. Further, the distributed controller provides a sense of robustness towards measurement noise [MR09]. Those benefits are the motivation for choosing

distributed control.

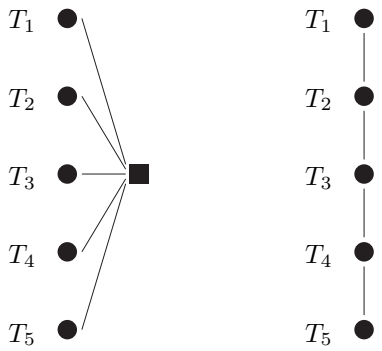

Figure 2.4: Example of centralized control (left) and a distributed control (right) in a wind farm of five wind turbines. In the centralized case, all wind turbines have two way communication with a separate wind farm controller. In the distributed case, the controller is distributed onto all the turbines in the farm. In this example, the turbines are allowed to communicate with the neighbors.

2.3 Thesis Overview

This thesis is divided into two parts.

- **Part I: Modular Distributed Wind Farm Control.**
  In this part of the work, a distributed wind farm controller is described. The focus of the controller design is modularity and scalability. The functionality and performance of this distributed controller is evaluated in a realistic simulation environment consisting of 10 wind turbines.
- **Part II: Distributed Controller for Thanet Wind Farm.**
  In this part of the work, an alternative control strategy is derived. The focus of the control strategy is that the controller must be implementable on the wind turbines at Thanet wind farm. This part of the work is the background for upcoming feedback control experiments at Thanet wind farm expected to take place in summer 2011.

An overview for each of these two parts of the thesis is presented in the following.

2.3.1 Part I: Modular Distributed Wind Farm Control

The first part of this work is based on [MMR11] and describes an alternative to the current static power distribution used in wind farms. The focus on this part of the work is modularity and scalability.

The modular and scalable controller developed in this part of the work assumes without loss of generality, that the wind turbines in the farm are placed in a row formation, as illustrated in Figure 2.5. Each wind turbine in the row is allowed to communicate with a given number of neighboring turbines.

As illustrated in the figure, each wind turbine is controlled by a power demand P^i_{dem}. Based on this and the incoming wind speed v^i, each wind turbine produces some power P^i_{out} and experiences

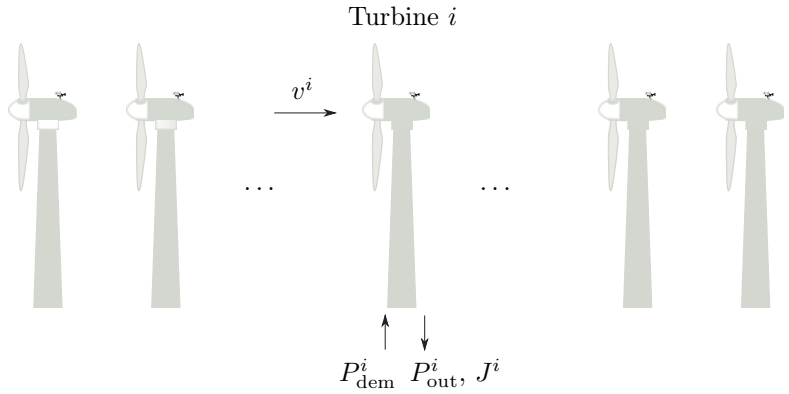

Figure 2.5: The assumed setup of the wind farm, with one row of turbines. Each turbine i experiences the incoming wind v^i and the power demand to the turbine P^i_{dem} as inputs. The outputs are produced power P^i_{out} and a measure of fatigue J^i on the turbine.

some fatigue J^i. The job of the wind farm controller is to ensure that the farm power demand is tracked, *i.e.* that $\sum_{i=1}^{N} P^i_{\text{out}} = P_{\text{farm,dem}}$ while the total fatigue $\sum_{i=1}^{N} J^i$ is minimized.

Based on [MMR11], this work describes how this fatigue minimization can be achieved using a distributed controller. Further it is shown that this controller can be designed such that it is both modular and scalable. Finally, this work examines the resulting wind farm controller in a realistic simulation environment, consisting of 10 National Renewable Energy Laboratory (NREL) wind turbines.

As mentioned, the focus of this controller design is to achieve modularity and scalability [MMR11].

- Scalability
  The control law on each turbine does not depend on the number of turbines in the wind farm. To add or remove a turbine in the wind farm, only the software on the neighboring turbines must be changed, not the entire wind farm. Moreover, the computational effort on each wind turbine in the farm does not depend on the number of turbines in the wind farm.

- Modularity
  The control law software on each turbine is identical. Only the communication path is different in each wind turbine.

2.3.2 Part II: Distributed Controller for Thanet Wind Farm

The topic of the second part of this work, is controller design for upcoming wind farm experiments at Thanet offshore wind farm in the UK. This is a part of the research project Aeolus [Aeo08] funded by the EU and completed in cooperation with Vestas. It is planned that two rounds of experiments are to be conducted, the first in the summer 2011 and the second in the fall of the same year.

One of the experiments to be conducted in the wind farm, is feedback control. This is the topic of this part of the thesis.

The goal is to develop a wind farm controller that dynamically updates the power set-points for the turbines in the wind farm, so that the wind farm power demand is tracked, while minimizing the fatigue on the turbines in the farm.

The controller must be implementable the Supervisory Control And Data Acquisition (SCADA) server at Thanet wind farm. As the wind turbines at Thanet are not designed for this type of

control, some very strict limitations are present.

For this reason, the focus on this second part of the work is to examine and accommodate the limitations of the SCADA server. Further it is desired that the controller is simplified as much as possible, while still illustrating the benefit of wind farm control.

In this second part of the work, we therefore simply work with one single standalone turbine, varying its power set-point in order to achieve fatigue reduction. The goal is to show, that fatigue reductions can be achieved by using the freedom that lies in the distribution of power set-points.

As the experiments are not yet conducted, the results of the experiments are not presented in this work.

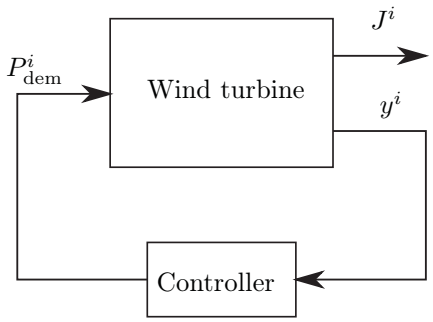

Figure 2.6: The simplified feedback controller for a standalone turbine, designed for the experiments at Thanet wind farm. The fatigue J^i is minimized by control of the input P^i_{dem}.

Part I

Modular Distributed Wind Farm Control

Chapter 3

Outline Part I

This chapter describes the outline of this first part of the Thesis. This part of the work modifies and evaluates the wind farm controller described in [MMR11].

- **Modeling**, Chapter 4
  This chapter describes how a dynamic wind farm model is obtained for a farm consisting of a number of National Renewable Energy Laboratory (NREL) wind turbines. Further it is examined how wind turbine fatigue can be modeled.
- **Wind Farm Problem Formulation**, Chapter 5
  This chapter describes how a linear wind turbine model can be obtained through linearization of the dynamic non-linear wind turbine model. Similarly it is described, how a quadratic function can be used to describe the wind turbine fatigue.
- **Modular Distributed Wind Farm Controller**, Chapter 6
  This chapter describes how a modular and scalable wind farm controller can be developed based on the linearized model, and by letting the wind turbines communicate with a limited number of neighbors.
- **Controller Performance Evaluation**, Chapter 7
  This chapter evaluates the performance of the modular and scalable wind farm controller, by simulating the performance in a wind farm consisting of 10 NREL wind turbines.

Chapter 4

Modeling

This chapter seeks to construct a model of a wind farm, useful for controller design. This includes wind turbine modeling, wind field modeling and fatigue modeling.

4.1 Wind Turbine Model

This section describes the wind turbine used as basis of the controller design throughout the following chapters.

4.1.1 The NREL Wind Turbine

It is chosen to base the wind farm controller design and later simulations, on the *National Renewable Energy Laboratory (NREL) offshore 5-MW baseline wind turbine*. A thorough model of this wind turbine has been constructed and is available [JBMS09], along with simplifications of the model [SJBMV07], [GSK+10]. The motivation for choosing this model as vantage point for the controller design, is that this model is designed to be representative of typical land- and sea based multimegawatt turbines [JBMS09].

Note further, that it is trivial to design the controller for another type of wind turbines, simply by replacing the NREL model in the controller design phase.

The NREL wind turbine, is a 5 MW variable speed offshore wind turbine with active hydraulic pitch control. In the following, an overview will be given of the NREL wind turbine, to provide the necessary understanding of the dynamics of the wind turbine. Note that this model is presented on an overview level, as the focus of this work is wind farm control rather than wind farm modeling. For more detail on the model of the NREL wind turbine, refer to [JBMS09]. Also note, that [SJBMV07] is the background for the modeling of the NREL wind turbine.

Note that the following model description, including data, is taken from [SJBMV07], [JBMS09], [Aeo10b] and [Aeo10a]. Parts of the scripts used in the sequel are likewise provided by the authors of [SJBMV07]. All source code is found in the enclosed CD, and the paths to the different scrips are presented in the following.

4.1.2 Model Purpose

The purpose of the model of the wind turbine, is to achieve a model useful for controller design at wind farm level. The model we are seeking is thus a closed loop wind turbine model with a

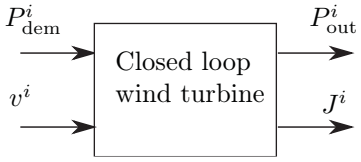

Figure 4.1: A wind turbine i seen as an actuator with controllable input P^i_{dem} and uncontrollable input v^i. The outputs are the actual power production P^i_{out} and some measure of fatigue J^i.

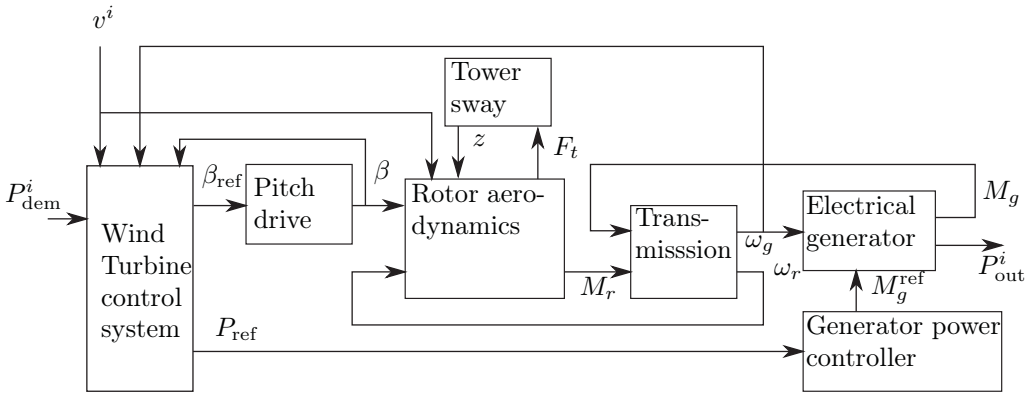

Figure 4.2: Overview of the components of a wind turbine [SJBMV07, p. 7].

local wind turbine controller.

On wind farm level, each wind turbine i is seen as an actuator with the power demand to the turbine P^i_{dem} as the controllable input, and the incoming wind v^i as an uncontrollable input. The output is the produced power P^i_{out} along with some measure of fatigue on the turbine J^i. This idea is illustrated in Figure 4.5.

We are therefore seeking a closed loop model of a wind turbine, *i.e.* a wind turbine with a local commercial controller. The local controller of the wind turbine assures that the turbine tracks the power set-point, while it minimizes various wind turbine loads. Such a model is developed in [SJBMV07], and is presented in the following.

The purpose of the wind farm controller designed in this work is not to compete with the local wind turbine controller, but to distribute power set-points to the turbines in the farm at a higher level. This means that the wind farm controller operate with relatively large time constants. The model of the NREL wind turbine to be used in wind farm controller design, needs therefore only to be accurate at low frequencies.

4.1.3 Wind Turbine Subsystems

In the following, the general components of the NREL wind turbine will be described based on Figure 4.2.

The following description of the components of the wind turbine is based on [SJBMV07, p.8-16], while the equations are based on [Aeo10a]

Rotor Aerodynamics

The model of the rotor dynamics, describes how the wind turbine rotor captures energy from the wind. The lift and drag on the blades of the turbine make the rotor turn generating a driving torque M_s, which is applied to the transmission system. The driving torque M_s depends on the incoming wind v and on the pitching of the blades of the turbine β, as illustrated in Figure 4.2.

The lift and drag on the wind turbine blades, also generates a force perpendicular to the plane swept by the blades. This is the thrust force F_t, causing the wind turbine tower to bend and thus to sway.

We describe the aerodynamics by the following two static equations.

$$\begin{aligned} M_s &= \frac{\pi}{2\omega_r}\rho R^2 v^3 C_P(\lambda,\beta) \\ F_t &= \frac{\pi}{2}\rho R^2 v^2 C_T(\lambda,\beta) \end{aligned}$$

where C_P is the power coefficient, and C_T is the thrust coefficient, both nonlinear wind turbine specific functions. The parameter ρ is the air density, v the effective incoming wind velocity [m/s], R the rotor radius [m], w_r is the rotor angular velocity [rad/s], λ is the tip speed ratio $\lambda = \frac{\omega_r R}{v}$ [-] and β is the pitch angle [°] [Aeo10a].

Transmission System

The transmission system contains the gearbox between the rotor shaft (low speed shaft) and the shaft between gearbox and generator (high speed shaft). In the modeling of the transmission system, the speed reduction in the gearbox must be taken into account. Also the stiffness and damping of the rotor shaft must be included in the model. The output is the angular velocity of the rotor ω_r and the angular velocity of the generator ω_g, and they depend on the rotor torque M_s generated by the wind, and on the applied generator torque M_g.

We model the transmission system as a 3rd oder drive train model based on a gear ratio, stiffness and damping as follows.

$$\begin{aligned} \omega_r &= \frac{1}{I_r}(M_s - \theta K_s - \dot{\theta} B_s) \\ \omega_g &= \frac{1}{I_g}(-M_g + \frac{1}{N}(\theta K_s + \dot{\theta} B_s)) \\ \dot{\theta} &= \omega_r - \frac{1}{N}\omega_g \end{aligned}$$

where ω_g is the angular velocity of the generator [rad/s], I_r and I_g are, respectively, the inertia of the rotor and the generator [kg/m^2], and θ [rad] is the torsion of the shaft. The constant K_s is the spring constant of the shaft [N/m], while B_s is the viscous friction constant of the shaft [N s/m], and N is the gear ration [-] [Aeo10a].

Electrical Generator

The electrical generator transforms the mechanical energy to electrical energy. Depending on the angular velocity of the generator ω_g and on the generator torque, the electrical generator will produce a given amount of output power P_{out}. The electrical generator is connected to the grid, and allows wind turbine operation at variable rotor speed.

Generator Power Controller

The generator power controller makes the wind turbine track the power set-point P^i_{dem} by controlling the reference signal for the turbine generator torque M_g^{ref}. By increasing the generator torque M_g, the power production will increase, while the angular velocity will remain the same.

We describe the electrical generator together with the generator power controller as a 1st order model with the input being P_{ref}. In other words, we do not model the generator torque reference M_g^{ref} explicitly. We use the following model.

$$\dot{M}_g = \frac{1}{\tau_g}\left(\frac{P_{\text{ref}}}{\omega_g} - M_g\right)$$

where τ_g is the generator time constant [s] [Aeo10a].

Tower Sway

The thrust force caused by the wind on the rotors, will excite the wind turbine tower. This will lead to an undesired nodding of the tower, producing fatigue to the wind turbine. Also, this nodding affects the rotor dynamics, as the relative wind speed will change as the turbine moves. It is a task of the local wind turbine controller, to minimize this tower sway.

We model the tower deflection p_{nac} [m], corresponding to the nacelle position, as a spring damper-system as follows.

$$\ddot{p}_{\text{nac}} = \frac{1}{m_t}\left(F_t - K_t p_{\text{nac}} - B_t \dot{p}_{\text{nac}}\right)$$

where m_t is the mass of the tower and K_t and B_t are, respectively, the spring and damper constants of the tower [Aeo10a].

Pitch Drive

In the NREL wind turbine, the actuators are hydraulic pitch actuators, with some given bandwidth. Therefore the pitch drive dynamics must be included in the wind turbine model, as the path from the pitch reference β_{ref} to the pitch angle of the blades β is characterized by a delay, first order filter characteristics and quantization.

The pitch drive is modeled as a second order system with a time constant τ_β and an input delay t_β, closed loop with a proportional controller with constant K_β.

$$\begin{aligned} \ddot{\beta} &= \frac{1}{\tau_\beta}(u_\beta e^{-t_\beta s}/s - \dot{\beta}) \\ u_\beta &= K_\beta(\beta_{\text{ref}} - \beta_{\text{meas}}) \end{aligned}$$

where β_{meas} is the measurement of the pitch [°] [Aeo10a].

Wind Turbine Controller

The two inputs considered to the wind turbine, is the effective wind velocity *vi* and the power demand P^i_{dem}. Here P^i_{dem} is the control variable of the wind farm controller designed in this work. The goal of the local wind turbine controller is to ensure that power set-point is tracked, *i.e.* that $P^i_{\text{out}} = P^i_{\text{dem}}$. When the power demand is not achievable, due to too low available power in the wind, the wind turbine controller must maximize the power production. This reveals two different modes of the wind turbine, *power maximization* and *power tracking*. We notice, that in

the power maximization mode, the wind turbine no longer acts as an actuator through P^i_{dem}, but solely tries to maximize the output power P^i_{out}.

The local wind turbine controller must additionally guarantee, that the turbine obeys the given constraints to the wind turbine, *e.g.* the speed and torque on the generator. Similarly, the local wind turbine controller must reduce the fatigue on the wind turbine, by damping the wind turbine tower bending and the shaft torsion.

It is worth noticing, that two internal control variables are the wind turbine pitch β and the reference to the generator torque M^{ref}_g, which provides the freedom to control the wind turbine in a desired manner.

The behavior in the two different control regions is.

- **Power Maximization**. In the power maximization, the pitch reference is zero $\beta_{\text{ref}} = 0$. At the same time, the generator torque reference is affected through the power reference P_{ref}, to maximize the power output P^i_{out}. The wind turbine controller does this by measuring the generator angular velocity ω_g, and by looking up the optimal generator torque M_g for the given angular velocity ω_g, and modifying P_{ref} accordingly.
- **Power Tracking**. In the power tracking mode, the wind turbine controller makes the wind turbine generator rotate with the nominal angular velocity $\omega_g = \omega^{\text{nom}}_g$, by measurements of ω_g and utilization of the pitch actuator, through β. It corrects the power set-point P^i_{dem} to compensate for transmission losses, and provides it as the power reference P_{ref}.

The controller is implemented in the following way. In the power maximization region, the power reference P_{ref} is simply found in a look-up table dependent on the generator angular velocity ω_g, while the pitch is kept constant at zero.

In the power tracking region, the generator power is kept constant at the desired power, while the turbine is brought to rotate with the nominal angular velocity using the blade pitching. This control system is implemented with a gain scheduled PI controller as follows.

$$\begin{aligned} \beta_{\text{ref}} &= K_P(\beta)\omega_e + K_I(\beta)\int_0^t \omega_e dt \\ K_P(\beta) &= K_{P,0}(\beta)\frac{\beta_2}{\beta_2+\beta} \\ K_I(\beta) &= K_{I,0}(\beta)\frac{\beta_2}{\beta_2+\beta} \end{aligned}$$

where ω_e is the error between the desired angular velocity of the generator, and the actual angular velocity of the generator. The proportional and integral gains K_P and K_I are based on the proportional and integral base gains $K_{P,0}$ and $K_{I,0}$ when there is no pitching $\beta = 0$, while the parameter β_2 is the pitch angle where the pitch sensitivity is doubled [Aeo10a].

4.1.4 Model Evaluation

To illustrate the dynamics of the model, some transient responses are presented in the following. The turbine is operating in the power tracking region, tracking a power set-point of 4 MW, and a wind field as illustrated in to the left in Figure 4.3 is applied. These inputs excite the turbine to produce the power illustrated in the left of Figure 4.3. In Figure 4.4 the resulting shaft torsion θ and tower deflection p_{nac} is illustrated. For model and script, refer to

`matlab/nrel/nrel_response.m`

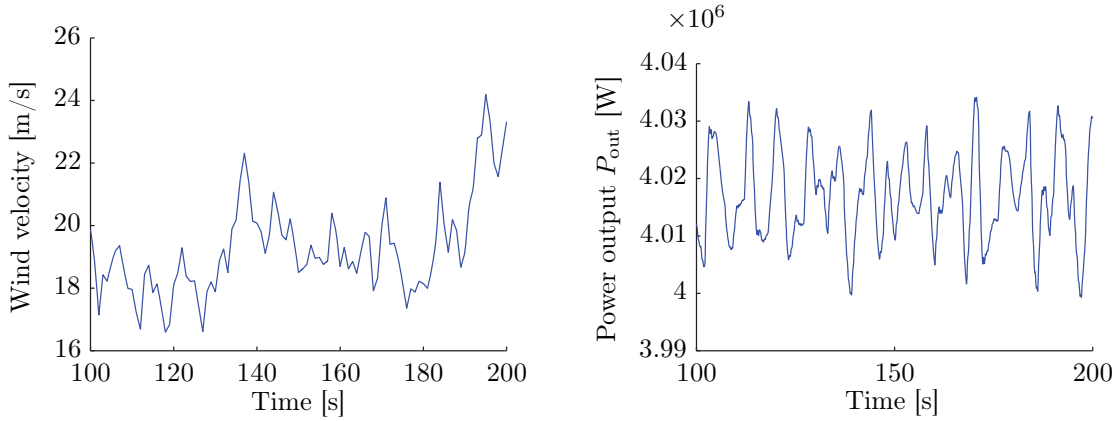

Figure 4.3: The wind field applied to the wind turbine (left), and the resulting power output (right), when the power demand P_{dem} to the turbine is 4 MW.

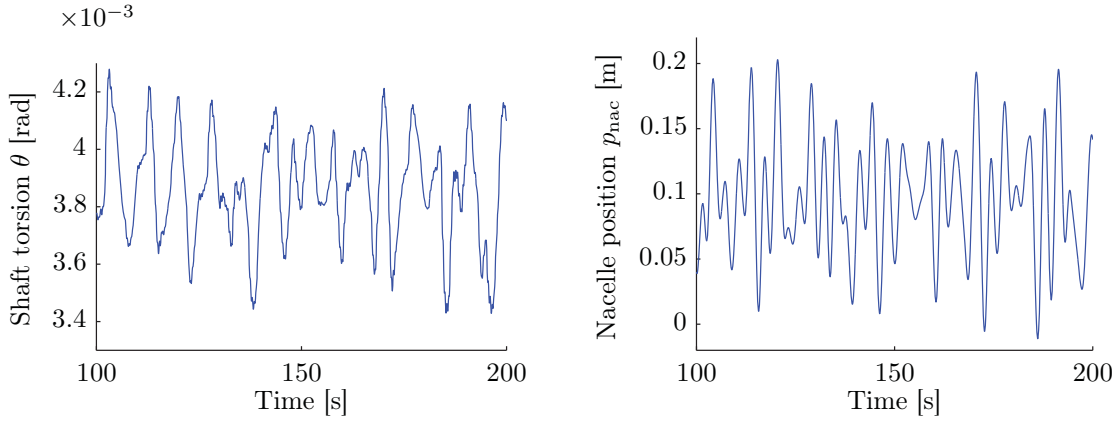

Figure 4.4: The torsion of the shaft (left) causing fatigue on the wind turbine generator, and the tower deflection (right) causing fatigue on the turbine structure.

4.2 Wind Farm Model

We consider a wind farm consisting of N NREL wind turbines. At wind farm level, we want the farm to produce the demanded power, while minimizing the fatigue on the turbines in the farm, *i.e.* we desire $\sum_{i=1}^{N} P_{\text{out}}^i = P_{\text{farm,out}}$ while minimizing $J_{\text{farm}} = \sum_{i=1}^{N} J^i$.

In the wind farm, we can consider each wind turbine as an actuator with the power demand P_{dem}^i and the incoming wind speed v^i as inputs. The power demand is a controllable input, while the wind is uncontrollable. The outputs of each actuator is the power produced P_{out}^i and the fatigue on the given wind turbine J^i. We construct a wind farm as N such actuators. Figure 4.1 illustrates a single turbine actuator.

Note that the wind turbine will only act as an actuator with a control variable, as long as the wind turbine is in the power tracking region, as is evident from last section. If the turbine is in the power maximization region, the wind turbine will only seek to maximize the power output, and will therefore be a disturbance in the wind farm model, as the power output P_{out}^i and fatigue output J^i will be uncontrollable.

Based on this, we construct a wind farm model by concatenating N independent wind turbine

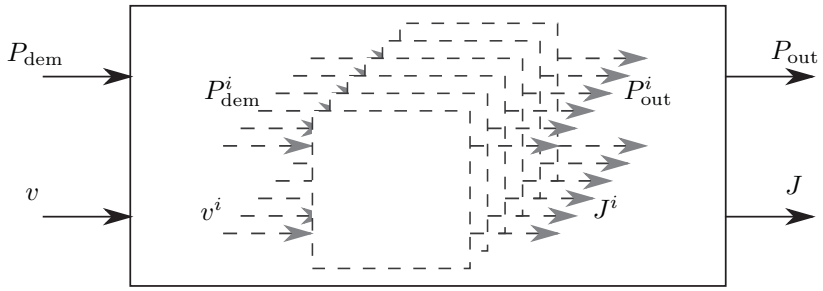

Figure 4.5: The wind turbine farm consisting of N wind turbines.

actuators, as illustrated in Figure 4.5. Each wind turbine is of the form previously presented, and seen in Figure 4.1. This means that we model the wind farm as a MIMO system with controllable input $P_{\text{dem}} \in \mathbf{R}^N$ and uncontrollable input $v \in \mathbf{R}^N$. The outputs of the wind farm model is the fatigue $J \in \mathbf{R}^N$ and the output power $P_{\text{out}} \in \mathbf{R}^N$.

Note that this means that it is chosen to neglect the coupling of wind turbines through wake. Obviously the wind turbines are coupled by the wind flow, as each wind turbine creates a wind deficit affecting downwind turbines [KBS09], [SKB09]. However it is unclear in what extend this coupling can be exploited for control purpose, *i.e.* if pitching one turbine in the farm can generate a desired response several hundred meters downwind. Current work seek to examine this coupling of wind turbines through the wind flow through wind farm experiments [Aeo11], and these results may be implemented in later work.

A second motivation for neglecting the coupling through the wind flow is that this will make the attempt to design a wind farm controller cleaner and simpler, while still showing the concept of wind farm control.

4.3 Wind Model

The incoming wind acts as an uncontrollable input v_i to each turbine in the wind farm. It is desired to use as much knowledge about the wind field as possible, in the controller design. Therefore we develop a wind model with white noise signal w_i as input, and the wind speed v_i as output.

It is useful to think of the wind as consisting of a mean wind speed $\overline{v}$ superimposed with turbulence fluctuations $\tilde{v}$. The mean wind speed $\overline{v}$ depends on the weather conditions, and varies on a time scale of several hours, while the wind turbine fluctuations vary on a quicker time scale but has zero mean when averaged over about 10 minutes [BSJB01, p. 17]. We thus write

$$v = \overline{v} + \tilde{v}. \tag{4.1}$$

The turbulence is typically generated because of thermal conditions, *e.g.* variations in temperature, and because of friction with the earth's surface. The turbulence is characterized by the turbulence intensity, defined as

$$T_I = \frac{\sigma_v}{\overline{v}} \tag{4.2}$$

where σ_v is the standard deviation of the wind speed variations around the mean wind speed $\overline{v}$. The wind speed variations can be assumed roughly Gaussian. [BSJB01, p. 17].

The turbulence spectrum can be described by describing its frequency content. This is commonly

done by the Kaimal spectrum [BSJB01, p. 23], [MMR11]

$$\Phi_v(\omega) = \sigma^2 \frac{4L_u/\overline{v}}{(1+\omega\frac{3L_u}{\pi\overline{v}})^{5/3}} \tag{4.3}$$

where ω is the frequency of the wind speed variations [rad] and L_u is a length scale [m].

4.4 Fatigue Model

The purpose of the wind farm controller is to avoid unnecessary fatigue on the wind turbines in the farm. It is therefore necessary to know what causes fatigue on a wind turbine. In the following, we base our fatigue model on the assessment deliverable [Cou08].

4.4.1 Components Causing Fatigue

During operation, the wind turbine will experience fatigue on the whole wind turbine structure, including structure, blades, gearbox etc. In the following, we will follow the assessment deliverable [Cou08], and only consider fatigue on the tower structure due to tower deflection, and on the gearbox shaft, due to torsion of the shaft. Details on this follow in the problem formulation in next section.

4.4.2 Measure of Fatigue

As suggested by the assessment deliverable [Cou08], it is chosen to use rainflow counting to evaluate the fatigue on the wind turbines. The ASTM standard [AST05] is used as the algorithm for rainflow counting in this work. In the following, we describe the concept of the rainflow algorithm.

Cyclic Stress

The main factors in fatigue due to varying loading is the stress amplitude and the number of stress cycles. In fatigue analysis, S-N curves are used to illustrate the relationship between those two quantities, by plotting the stress level S_a versus the number of cycles before structure failure N_f. The relationship is typically plotted in a log-log plots, as the relationship has characteristics close to

$$S_a(N_f) = AN_f^B, \tag{4.4}$$

where A and B are constants [SF01, p. 68].

A S-N curve for a given component can therefore tell how many cycles of the exact same stress amplitude can be applied to the component, before failure. The stress on a wind turbine will however consist of many different amplitudes. Therefore it is desired to be able to evaluate the fatigue caused by different stress amplitudes each with a given number of stress cycles. Miner's rule describe exactly this, and simply states that the damage done by a series of different stress amplitudes $S_{a,i}$ with corresponding number of cycles n_i and corresponding number of cycles before structure failure $N_{f,i}$, $i = 1, ..., m$, can be summed up to give the total damage [SF01, p. 275]

$$D = \sum_{i=1}^{m} \frac{n_i}{N_{f,i}}. \tag{4.5}$$

Here it is defined that the damage done by one cycle with stress level N_f is $D = \frac{1}{N_f}$.

Rainflow Counting Algorithm

Based on Miner's rule Equation 4.5, it is possible to calculate the damage D on a wind turbine, when the relationship $S_a(N_f)$ is known, and when the stress cycles are counted and the stress amplitudes are measured.

However, a method is needed in order to extract stress cycles and amplitudes from a time varying signal. Amplitudes and cycles are obvious when the signals are sinusoids, but not when the signal is a summation of various signals. Figure A.1 (right) in appendix A on page 72 illustrates such a signal, where cycles and amplitudes are not obvious. In this work the rainflow counting algorithm is used for this purpose, as this method is widely used in the literature [BSJB01], [NCVT10], [Ham06]. Also the assessment deliverable [Cou08] suggests the use of rainflow counting.

In Appendix A on page 72, the rainflow counting method is described, based on [AST05] along with an example, in order to enhance the understanding of the algorithm.

In the following chapter, we will show how these models of the turbine, the wind field and the fatigue on the turbine, can be used to make a problem formulation of the wind farm optimization.

Chapter 5

Wind Farm Problem Formulation

In this chapter, we describe the objective of the wind farm control, based on a performance assessment deliverable [Cou08]. We then reformulate this objective into a quadratic problem based on the wind farm model and the fatigue model. This is done by linearizing the wind field model and the wind turbine model, and by forming a quadratic description of the wind farm performance.

5.1 Initial Problem Formulation

In the following we use the assessment deliverable [Cou08] concerning wind turbine fatigue, to to state an initial problem formulation for wind farm control.

5.1.1 Wind Farm Performance Description

The objective of the wind farm controller is to enhance the overall wind farm performance. Ideally this performance would include both power efficiency, power quality and a large number of mechanical loads including extreme loads, fatigue loads and thermal loads [Ham06], which in practice is not sensible [Cou08]. Therefore it proposed in [Cou08] to use the following three terms as basis of a measure of performance.

1. Deviation from the power reference, described by the scalar cost J_{power}.
2. Fatigue load due to tower oscillation, described by the scalar cost J_{tower}.
3. Fatigue load due to shaft torsion, described by the scalar cost J_{shaft}.

The reason for choosing just these three measures of performance, is that a larger number of components will increase the modeling complexity, but not lead to a different type of control problem [Cou08].

Deviation From Power Reference

The performance of the power is measured by using the RMS value of the deviation from the farm power demand [Cou08]

$$J_{\text{power}}(P_{\text{farm,out}}) = \sqrt{\frac{1}{T}\int_{t=0}^{T} (P_{\text{farm,out}} - P_{\text{farm,dem}})^2 dt} \tag{5.1}$$

where $P_{\text{farm,out}} = \sum_{i=1}^{N} P_{\text{out}}^i$ [W] is the total power output of the N turbines in the wind farm, while $P_{\text{farm,dem}}$ [W] is the total power demand to the wind farm.

Fatigue Load - Tower Oscillation

The fatigue caused by the tower oscillation for each turbine, is based on the position of the wind turbine nacelle p_{nac} [m]. Rainflow counting combined with Miner's rule, as described in Section 4.4, is used to calculate the generated damage on each turbine. This damage is summed up over all the turbines, to provide a measure of tower oscillation fatigue in the whole park. We write this as follows.

$$J_{\text{tower}}(p_{\text{nac}}) = \sum_{i=1}^{N} \mathcal{R}(p_{\text{nac}}^i) \tag{5.2}$$

where p_{nac}^i is the position of the nacelle of wind turbine i. Here $\mathcal{R}(\cdot)$ denotes the rainflow algorithm, as described in Section 4.4.2.

Fatigue Load - Shaft Torsion

The fatigue on the shaft is based on the torsion of the shaft, θ [rad]. Again rainflow counting combined with Minter's rule is used to calculate the generated damage on the shaft in each turbine. This damage is summed up over all turbines, to provide a measure of the shaft torsion fatigue in the whole park. We write this as follows.

$$J_{\text{shaft}}(\theta) = \sum_{i=1}^{N} \mathcal{R}(\theta^i) \tag{5.3}$$

where θ^i is the angle of the nacelle of wind turbine i.

Total Performance Function

Based on the above, we can describe the total performance $J \in \mathbf{R}$ [Cou08] as

$$J(P_{\text{farm,out}}, p_{\text{nac}}, \theta) = [J_{\text{power}},\ J_{\text{tower}},\ J_{\text{shaft}}]c \tag{5.4}$$

where $c \in \mathbf{R}^3$ is some scalarization parameter.

5.1.2 Problem Formulation

We can now state the problem formulation for the wind park controller design, based on the assessment deliverable.

The objective of the wind farm controller, is to minimize J, subject to the dynamics and constraints of the wind farm. The optimization variables are the wind turbine set-points $P_{\text{dem}} \in \mathbf{R}^N$. Further, the controller must follow the concept of distributed control. It is assumed that the turbines are located in a row formation, and the turbines are only allowed to communicate with a given number of neighbors.

In the following, approximations of the wind farm dynamics and the fatigue model will be conducted, such that the problem formulation can be stated as a quadratic optimization problem with constraints to the communication structure.

5.2 Linear Wind Farm Description

For controller design purpose, a linear model of the wind farm is needed, which means that a linear wind turbine model and a linear wind field model must be derived. This is the content of this section. The wind turbine linearization is taken from [SJBMV07], while the wind model linearization is based on [BSJB01].

5.2.1 Linear Wind Turbine Description

A linear model of the National Renewable Energy Laboratory (NREL) wind turbine described in Section 4.1 on page 16 is needed for design purpose. The model should be linearized around an operating point dependent on the inputs v^i and P^i_{dem}.

Such a model is developed as a part of the Aeolus project. A detailed description of the linearization can be found in [SJBMV07]. The linearization script is found in

`matlab/nrel/NREL/`

This linear model is developed with farm control as focus. The input to the model is therefore the power set-point for the turbine and the uncontrollable wind input. The outputs are chosen to be the torque on the shaft and tower, as reducing these is believed to reduce fatigue on the turbine. This is described further in the sequel.

As the model is developed for farm control, this also means that a number of simplifications can be made, in order to reduce the model complexity. These simplifications can be made, as the wind farm controller will operate at a slower sampling rate, than the local wind farm controller. This motivates removing high frequency content of the wind turbine.

For this reason, the oscillation of the tower and shaft is neglected. Also, the transmission filter dynamics are removed. The pitch actuator is also simplified, to being infinitely fast and linear [SJBMV07]. The end result is a 3rd order state space model of the wind turbine, when operating in the power tracking region. As only this region is of relevance for the farm control, only this model will be presented. For turbine i, the state space model is described as follows, with parameters as specified in [SJBMV07, p. 60].

$$\begin{aligned} \dot{x}_t &= A_t x_t + B_t u_t + B_{d,t} d_t \\ y_t &= C_t x_t + D_t u_t + D_{d,t} d_t \\ x_t &= [\beta^i,\ \omega_r^i,\ \omega_g^{\text{filt},i}]^T \end{aligned}$$

where the inputs and outputs are given as small signal values around operating points

$$\begin{aligned} u_t &= P^i_{\text{dem}} - \overline{P^i_{\text{dem}}} \\ d_t &= v^i - \overline{v^i} \\ y_t &= [M_t^i - \overline{M_t^i},\ M_s^i - \overline{M_t^i}]^T. \end{aligned}$$

Here $\overline{P^i_{\text{dem}}}$ is the operation point of the power demand, $\overline{v^i}$ the operation point of the incoming wind and $\overline{M_t^i}$ and $\overline{M_s^i}$ the tower and shaft torque operation points.

Similarly, the turbine state x_t also consists of small signal values such that

$$x_t = [\beta^i - \overline{\beta^i},\ \omega_r^i - \overline{\omega_r^i},\ \omega_g^{\text{filt},i} - \overline{\omega_g^{\text{filt},i}}]^T$$

where β^i is the small signal value of the turbine blade pitch, ω_r^i is the small signal value of rotor

angular velocity and $\omega_g^{\text{filt},i}$ is the small signal value of the filtered generator angular velocity. Each of these have corresponding operating point, as stated.

Note that the model reductions lead to the simplification that

$$P_{\text{out}}^i = P_{\text{dem}}^i. \tag{5.5}$$

This is due to the effect of the local turbine controller, which seeks this objective with faster dynamics, than what can be controlled at farm level.

5.2.2 Linear Wind Model

It is also desired to find a linear model of the wind field model presented in 4.3 on page 22. This linear wind model is constructed such that the input is white noise, while the output is the wind speed v^i.

The linear model is constructed by fitting a linear stable system G_v to the turbulence spectrum presented in Equation 4.3. We denote the linear model G_v, and represent it on state space form as

$$\begin{aligned} \dot{x}_v &= A_v x_v + B_v w_v \\ y_v &= C_v x_v \end{aligned} \tag{5.6}$$

with state x_v and white noise w_v with spectrum $\Phi_w(\omega) = 1$ as input. We here refer to [GL00, p. 108] concerning the notation and note, that as white noise in continuous time has infinite energy, this equation involves some idealization. In the sequel the system will be discretized, and the mathematical difficulties of infinite energy is avoided.

The goal is that the spectrum $\Phi_y(\omega)$ of the output y_v equals the turbulence spectrum $\Phi_v(\omega)$, *i.e.* $\Phi_y(\omega) = \Phi_v(\omega)$. To achieve this, it is known that the system G_v must satisfy [GL00, p. 112]

$$\Phi_v(\omega) = G_v(j\omega)\Phi_w(\omega)G_v^*(j\omega) = \left|G_v(j\omega)\right|^2. \tag{5.7}$$

An example of the Kaimal spectrum is depicted in Figure 5.1. The spectrum is fitted manually with pole-zero placement with a second order linear filter G, which is depicted in the same plot. The linear model is a 2nd order model and is found in:

`matlab/wind_filter/wind_filter.m`

Hereby we have a linear model of the wind field dynamics by Equation 5.6.

5.2.3 Linear Wind Farm Model

We construct the wind farm model with N turbines in three steps. First we incorporate the wind model into the wind turbine model. Secondly we concatenate N of these turbine models to obtain a model of an entire wind farm. Finally we perform a discretization of the wind farm model.

Combining the Wind Turbine and Wind Field Model

We can now concatenate the model of the wind field, with the model of the wind turbine. Hereby we obtain a model of the wind turbine, taking only white noise as uncontrollable input. This is

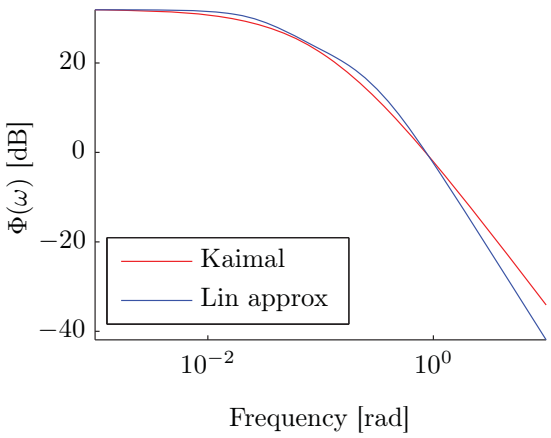

Figure 5.1: Second order linear approximation of the Kaimal spectrum, here with parameters $T_I = 0.1$, $\overline{v} = 15$ m/s, $L_u = 150$ m.

simply done by letting the output of the wind filter be the input to the turbine model.

$$\begin{aligned} \dot{x}_c &= A_c x_c + B_c u_t + B_{d,c} w_v \\ y_t &= C_c x_c + D_c u_t \end{aligned} \tag{5.8}$$

where

$$\begin{aligned} x_c &= [x_t,\ x_v]^T \\ A_c &= \begin{bmatrix} A_t & B_{d,t}C_v \\ 0 & A_v \end{bmatrix} \\ B_c &= \begin{bmatrix} B_t \\ 0 \end{bmatrix} \\ B_{d,c} &= \begin{bmatrix} B_{d,t}D_v \\ B_v \end{bmatrix} \\ C_c &= \begin{bmatrix} C_t & D_{d,t}C_v \end{bmatrix} \\ D_c &= D_t. \\ D_{d,c} &= D_{d,t}D_{d,v} \end{aligned}$$

We see that this concatenated wind turbine model only has the controllable input u_t and white noise w as inputs, as desired. Again notice that the continuous time white noise is to be seen as an idealization, which is resolved in the later discretization of the model.

Constructing the Wind Farm Model

To construct a model consisting of N wind turbines, we concatenate N systems as described by 5.8 and get the following wind farm model.

$$\begin{aligned} \dot{x} &= A_f x + B_f u + B_{d,f} w \\ y &= C_f x + D_f u \end{aligned} \tag{5.9}$$

where the system simply is obtained by block concatenation

$$\begin{aligned} A_f &= \mathbf{diag}(A_c, \ldots, A_c) \\ B_f &= \mathbf{diag}(B_c, \ldots, B_c) \\ B_{d,f} &= \mathbf{diag}(B_{d,c}, \ldots, B_{d,c}) \\ C_f &= \mathbf{diag}(C_c, \ldots, C_c) \\ D_f &= \mathbf{diag}(D_c, \ldots, D_c) \\ x &= [x_{c,1}^T, \ldots, x_{c,N}^T]^T \\ y &= [y_{t,1}^T, \ldots, y_{t,N}^T]^T \\ w &= [w_{v,1}, \ldots, w_{v,N}]^T. \end{aligned}$$

Each of the new wind farm model matrices contain N turbine model matrices. In this formulation all N turbine models are assumed identical. Inserting different turbine models can however readily be done.

Discrete Wind Farm Model

We are now ready to present the final wind turbine model, based on the above. The last step is to discretize the continuous wind turbine model described by Equations 5.9. This is done by zero order hold with a given sample time T_s. The wind farm model thus becomes

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + B_d w(k) \\ y(k) &= Cx(k) + Du(k). \end{aligned} \tag{5.10}$$

We now have a discrete linear representation of a wind farm consisting of N wind turbines. Each turbine has p states, so $A \in \mathbf{R}^{Np \times Np}$, $B,\ B_d \in \mathbf{R}^{Np \times N}$, $C \in \mathbf{R}^{N \times Np}$, $D \in \mathbf{R}^{p \times p}$.

The uncontrollable input due to wind $w(k)$ is characterized by the covariance matrix $W = \mathbf{E}(ww^T)$.

5.2.4 Evaluation of Linearized Model

It is desired to evaluate the performance of the linearized model. We do this by observing the behavior of a single NREL wind turbine.

The performance evaluation is performed by comparing a linearized NREL wind turbine model to the original NREL wind turbine model. This evaluation is conducted by applying a wind field similar to the one depicted in Figure 4.3 together with a power demand of 4 MW to both models. The linear model is linearized around the power demand operating point 4 MW and the wind speed operating point corresponding to the applied wind field, which is 20 m/s.

Figures 5.2 and 5.3 show four responses of the nonlinear and the discretized linear model. In this example a sample time of $T_s = 1$ is used. The figure illustrates that the linear model captures the dynamics of the nonlinear wind turbine model with sampling times as slow as 1 Hz.

5.3 Quadratic Fatigue Description

For controller design purpose, it is desired to have a quadratic cost function of the linearized turbine model. This is the subject of this section, where a measure of fatigue used in the literature is compared to the measure of fatigue base on the rainflow counting algorithm.

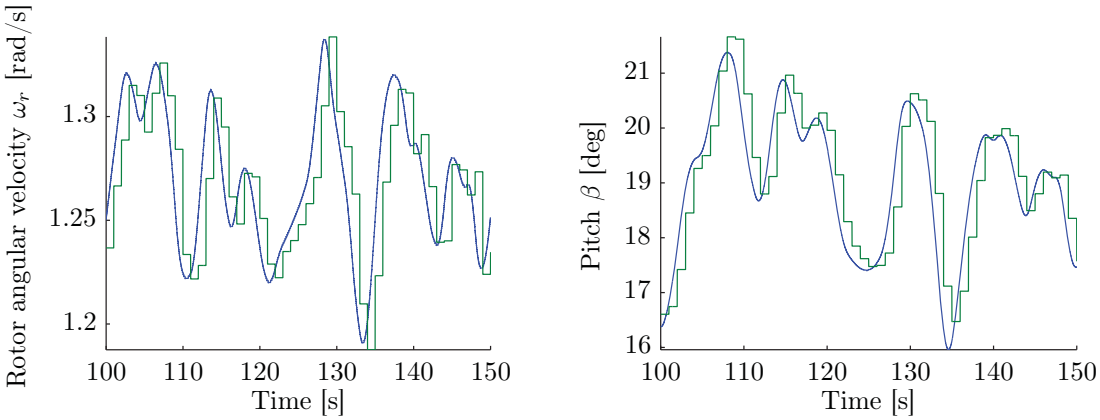

Figure 5.2: Comparison of the NREL model (blue) and simplified discrete model (green) for two of the turbine states.

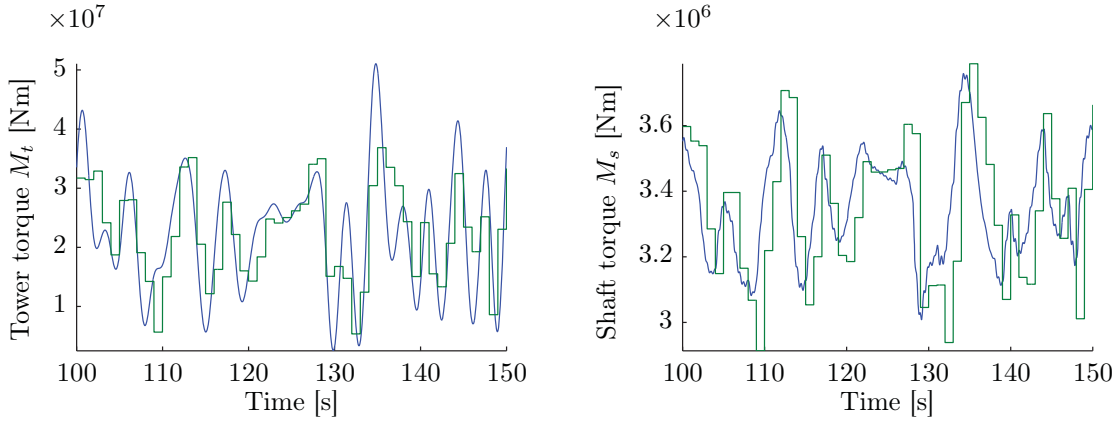

Figure 5.3: Comparison of the NREL model (blue) and simplified discrete model (green) for the outputs of the discrete model.

Quadratic Fatigue Model Proposal

As described, we measure fatigue as

$$\begin{aligned} J &= [J_{\text{power}},\ J_{\text{tower}},\ J_{\text{shaft}}]c \\ &= [J_{\text{power}},\ \mathcal{R}(p_{\text{nac}}), \mathcal{R}(\theta)]c \end{aligned}$$

where J_{power} is the RMS value of the deviation between desired power production and actual power production (Equation 5.1) The function $\mathcal{R}$ denotes the rainflow algorithm. The constant $c \in \mathbf{R}^3$ is a weighting parameter.

We first note, that based on the linear model, it is not possible to penalize the first term, which is the power tracking deviation. The reason is, that the linearization implies that $P^i_{\text{dem}} = P^i_{\text{out}}$. Instead of penalize this deviation, we instead require that the sum of the input signals is zero, *i.e.* $\mathbf{1}^T u = 0$, as this will yield $P_{\text{out}} = P_{\text{dem}}$ for the linearized model.

For this reason, we only penalize the scalars J_{tower} and J_{shaft}. We remind that p_{nac} is the position of the nacelle, so the scalar $J_{\text{tower}} = \mathcal{R}(p_{\text{nac}})$ is a measure of fatigue on the tower, while θ is the torsion of the shaft, so the scalar $J_{\text{shaft}} = \mathcal{R}(\theta)$ is a measure of fatigue on the shaft.

The rainflow algorithm is presented in Section 4.4.2, while the parameters to be used in the rainflow algorithm are presented in [Cou08]. Note that different parameters are used for the fatigue on the tower and fatigue on the shaft, such that

$$\begin{aligned} B_{\text{shaft}} &= 8 \\ B_{\text{tower}} &= 4 \end{aligned}$$

, see Equation 4.4.

As it is desired to have a quadratic measure of fatigue, we need an alternative to the measure based on the rainflow algorithm. In the literature, *e.g.* [Spu], [MMR11] it is suggested, that the torque on the tower and the torque on the shaft, can be used to model fatigue on tower and shaft respectively. In the presented literature, a quadratic function of the torque on the tower $\mathcal{Q}(M_t) = |M_t|_{Q_t}$ is used as a measure of fatigue on the tower, while a quadratic function on the shaft $\mathcal{Q}(M_s) = |M_s|_{Q_s}$ is used as a measure of fatigue on the shaft.

This proposes that we use the following quadratic measure of fatigue

$$J = \mathcal{Q}(M_t) + c_s \mathcal{Q}(M_s) \tag{5.11}$$

where $c_s \in \mathbf{R}$ is some appropriate weight, while imposing the constraint that $\mathbf{1}^T u = 0$.

In the following, this new expression for fatigue will be evaluated based on the NREL wind turbine model presented in [Aeo10b].

Evaluation of Quadratic Model

In the following we will compare the proposed quadratic fatigue model $\mathcal{Q}(M_s)$ and $\mathcal{Q}(M_t)$ with the rainflow counting based fatigue model $\mathcal{R}(\theta)$ and $\mathcal{R}(p_{\text{nac}})$. This will be done by evaluating the correlation between $\mathcal{R}(p_{\text{nac}})$ and $\mathcal{Q}(M_t)$ and the correlation between $\mathcal{R}(\theta)$ and $\mathcal{Q}(M_s)$. The goal is to verify that the quadratic model can be used as a measure of fatigue in the controller design.

The evaluation will be performed applying a control input u and wind input v to the NREL wind turbine model. Based on this we measure the fatigue using both the quadratic and the rainflow method. Every such simulation will provide the corresponding pair $(\mathcal{R}(p_{\text{nac}}),\ \mathcal{Q}(M_t))$ and the pair $(\mathcal{R}(\theta),\ \mathcal{Q}(M_s))$. By doing this a number of time, we can observe the correlation between the two different fatigue measures.

In the first evaluation, the power demand P_{dem} is kept constant, while wind fields of different turbulence intensities are applied. The turbulence intensity T_I is varied from 4 % to 20 %, as this covers the test specification in [SJBMV], which operates with turbulence intensities from 7 % to 10 %.

A sample of two such wind fields is presented in Figure 5.4, along with the corresponding tower bending p_{nac}. The tower bending obviously increases with increasing wind turbulence intensity.

The pairs $(\mathcal{R}(p_{\text{nac}}),\ \mathcal{Q}(M_t))$ and $(\mathcal{R}(\theta),\ \mathcal{Q}(M_s))$ for a number of such simulations, are illustrated in Figure 5.5.

In the second evaluation, the effective wind velocity v is kept constant, while the power demand P_{dem} is perturbed with noise with different standard deviation. This results in pairs $(\mathcal{R}(p_{\text{nac}}),\ \mathcal{Q}(M_t))$ and $(\mathcal{R}(\theta),\ \mathcal{Q}(M_s))$ as illustrated in Figure 5.6.

The details of the simulations can be found in :

`matlab/fatigue/fatigue.m`

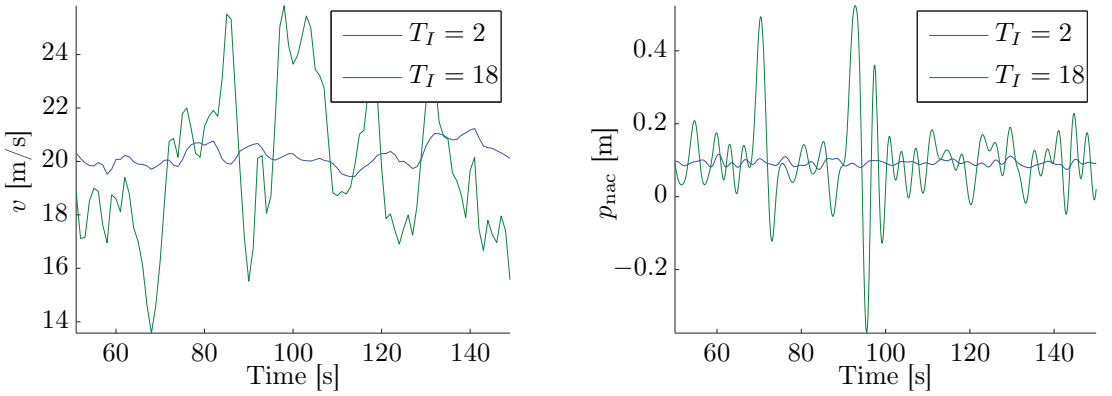

Figure 5.4: The two applied wind fields with different turbulence intensity (left) and the resulting tower bending (right).

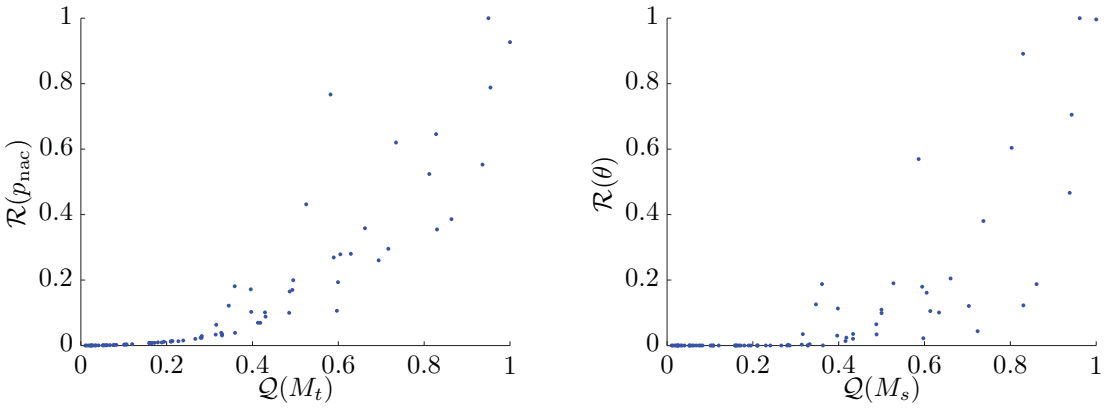

Figure 5.5: The relationship between fatigue measured by rainflow counting function $\mathcal{R}$ and by a quadratic function $\mathcal{Q}$. The pairs are obtained by applying different wind fields v. Note that the axes are normalized.

Conclusion

Both Figures 5.5 and 5.6 illustrate, that a reduction in $\mathcal{Q}(M_t)$ will reduce $\mathcal{R}(p_{\text{nac}})$, and that a reduction in $\mathcal{Q}(M_s)$ will reduce $\mathcal{R}(\theta)$. Based on this, we justify the use of the quadratic fatigue measures, which will be used in the controller design in the sequel.

Note that simulations with similar results are obtained with different operation points of the linear model.

Also note, that the seemingly more random values of $\mathcal{Q}(M_s)$, is a result of the large exponent $B_{\text{shaft}} = 8$. This large exponent makes the value of $\mathcal{Q}(M_s)$ highly dependent of just a single peaks of high amplitude, see Equation 4.4.

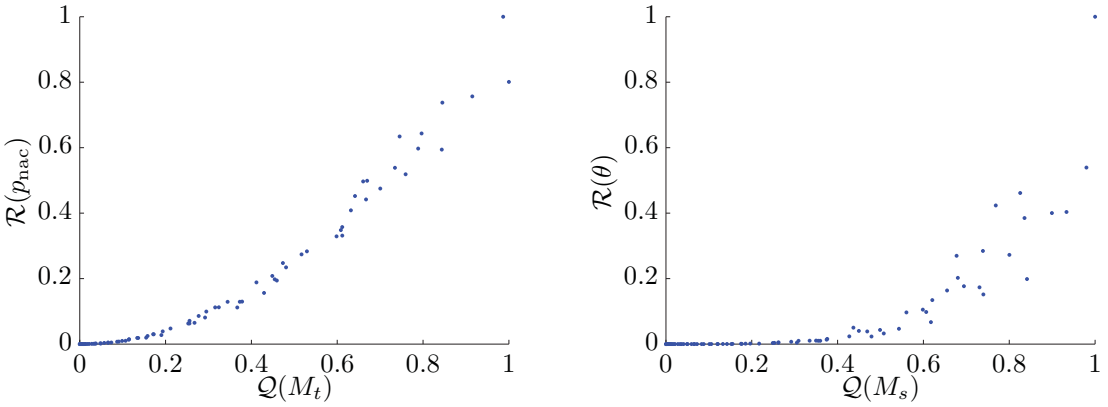

Figure 5.6: The relationship between fatigue measured by rainflow counting function $\mathcal{R}$ and by a quadratic function $\mathcal{Q}$. The pairs are obtained by perturbing the power demand to the turbine. Note that the axes are normalized.

5.4 Quadratic Problem Formulation

Based on the linear discrete wind turbine model and the quadratic cost function derived in this chapter, we can formulate the wind farm control problem as follows

$$\begin{array}{ll} \text{minimize} & J = \mathbf{E}(y^T Y y + u^T U u) \\ \text{subject to} & x(k+1) = Ax(k) + Bu(k) + B_d w(k) \\ & y(k) = Cx(k) + Du(k) \end{array} \tag{5.12}$$

with variables $u(k) \in \mathbf{R}^N$, $y(k) \in \mathbf{R}^{2N}$ and $x \in \mathbf{R}^{Np}$, as each of the N turbines has p states, two outputs and one input.

We now only need to incorporate the constraint that the wind farm produces the demanded amount of power. As described by 5.5, the dynamics from the power set-point to the produced power of a single turbine are too fast for wind farm control. We therefore simply require that the sum of the power set-points for the turbines must equal the farm power demand. As described in last section, this leads to the constraint that $\mathbf{1}^T u = 0$. We add this constraint to Problem 5.12 and get

$$\begin{array}{ll} \text{minimize} & J = \mathbf{E}(y^T Y y + u^T U u) \\ \text{subject to} & x(k+1) = Ax(k) + Bu(k) + B_d w(k) \\ & y(k) = Cx(k) + Du(k) \\ & \mathbf{1}^T u = 0 \end{array} \tag{5.13}$$

still with variables $u(k) \in \mathbf{R}^N$, $y(k) \in \mathbf{R}^{2N}$ and $x \in \mathbf{R}^{Np}$.

The matrices $Y, U \geq 0$, respectively, penalize the output and the input. This means that the matrix Y penalizes the shaft and tower torques M_s, M_t of the N turbines in the farm, while U penalizes the power set-points of the turbines.

In the following we perform three reformulations of the Problem 5.13. First we eliminate the output equation, and secondly we eliminate the global power constraint equation. Finally we include the requirement, that the control should be distributed.

Elimination of Power Output Equation

We reformulate the cost function to depend on x and u by letting

$$\begin{aligned} J &= \mathbf{E}(y^T Y y + u^T U u) \\ &= \mathbf{E}(x^T Q x + 2x^T S u + u^T R u) \end{aligned}$$

with $Q = C^T Y C$, $S = \frac{1}{2}(C^T Y D + C^T Y^T D)$, $R = D^T Y D + U$.

Eliminating the Global Power Constraint

To eliminate the global power constraint $\mathbf{1}^T u = 0$, we find a matrix $T \in \mathbf{R}^{N \times N-1}$ that parameterizes the linear feasible set [BV04, p. 537]:

$$\{u \mid Tu = 0\} = \{\hat{u} \mid \hat{u} \in \mathbf{R}^{N-1}\}. \tag{5.14}$$

We choose the matrix T such that [MMR11]

$$\begin{aligned} u_1 &= \hat{u}_1 \\ u_i &= \hat{u}_i - \hat{u}_{i-1}, \qquad i = 1, ..., N-1 \\ u_N &= -\hat{u}_{N-1}. \end{aligned}$$

which can be expressed as

$$u = T\hat{u} \tag{5.15}$$

The matrix T is 1 on the main diagonal and -1 on the first lower subdiagonal. Using $\hat{u}$ as optimization variable instead of u will guarantee that the desired power constraint $\mathbf{1}^T u = 0$ is honored due to the transformation matrix T.

In the sequel it will become evident, that this choice of T is appropriate for the chosen formation of the wind turbines.

We change the optimization problem according to the change of variables

$$\begin{aligned} x(k+1) &= Ax(k) + BT\hat{u}(k) + B_d w(k) \\ &= Ax(k) + \hat{B}\hat{u}(k) + B_d w(k) && (5.16) \\ J &= \mathbf{E}(x^T Q x + 2x^T S T \hat{u} + \hat{u}^T T^T R T \hat{u}) \\ &= \mathbf{E}(x^T Q x + 2x^T \hat{S}\hat{u} + \hat{u}^T \hat{R}\hat{u}) && (5.17) \\ & && (5.18) \end{aligned}$$

where we let $\hat{B} = BT$, $\hat{S} = ST$, $\hat{R} = T^T R T$.

It is noted, that the control signal $\hat{u}(k)$ readily is transformed to the original control signal as $u = T\hat{u}(k)$.

Distributed Feedback

Ignoring the requirement of distributed feedback, we can express the optimization problem as follows, based on the above reformulations.

$$\begin{array}{ll} \text{minimize} & J = \mathbf{E}(x^T Q x + 2x^T \hat{S}\hat{u} + \hat{u}^T \hat{R}\hat{u}) \\ \text{subject to} & x(k+1) = Ax(k) + \hat{B}\hat{u}(k) + B_d w(k) \end{array} \tag{5.19}$$

We know that the solution to a problem on this form is linear state feedback [ÅW97, p. 412]

$$\hat{u}(k) = -\hat{L}x(k) \tag{5.20}$$

with some given feedback matrix L. In terms of the original problem, the control law simply becomes

$$u(k) = -Lx(k) \tag{5.21}$$

where $L = T\hat{L}$.

The requirement, that the feedback must be distributed, can be expressed by imposing a structure of the feedback matrix L. We can describe this as $L \in \mathcal{L}$, where $\mathcal{L}$ ensures that the structure of L corresponds to the available state knowledge. This imposes a structure on $\hat{L}$ which we express as $\hat{L} \in \hat{\mathcal{L}}$. In the following, we describe this structural requirement $\hat{\mathcal{L}}$.

For the Problem 5.19 including the constraint 5.20, we recall that

$$x = [x_1^T, ..., x_N^T]^T, \quad u = [u_1, ..., u_N]^T$$

where $x_i \in \mathbf{R}^p$ is the state of agent i and $u_i \in \mathbf{R}$ is the control input of agent i.

We remind that a row formation of the turbines is assumed, where the turbines only are allowed to communicate with a given number of up- and downwind turbines. This means that agent i is allowed to calculate its respective control signal u_i from the state feedback $u = -Lx$ based solely on knowledge of its own state x_i and input u_i, along with given states x_j and input signals u_j from neighboring turbines, for some given set of j.

Based on this communication pattern, the structure imposed on $\hat{L}$ can be described as $\hat{L} \in \hat{\mathcal{L}}$ with [MMR11]

$$\hat{\mathcal{L}} = \{X \,|(X)_{i,j} \neq 0 \quad \text{only if} \quad i - l_1 \leq j \leq i - l_2\}, \tag{5.22}$$

where l_1 and l_2 are parameters determining the structure. The above means that the control input u_i to turbine i depends on $l_1 + 1$ upwind turbines and l_2 downwind turbines. This can be realized as $u_i = L_i x = (T\hat{L})_i x$, and by observing the structure of T.

Final Problem Formulation

With the above reformulations, the problem formulation becomes as follows.

$$\begin{array}{ll} \text{minimize} & J(\hat{L}) = \mathbf{E}(x^T Q x + 2x^T \hat{S}\hat{u} + \hat{u}^T \hat{R}\hat{u}) \\ \text{subject to} & x(k+1) = Ax(k) + \hat{B}\hat{u}(k) + B_d w(k) \\ & \hat{u}(k) = -\hat{L}x(k) \\ & \hat{L} \in \hat{\mathcal{L}} \end{array} \tag{5.23}$$

where the variables are $\hat{L} \in \mathbf{R}^{N-1 \times Np}$, $x \in \mathbf{R}^{Np}$ and $\hat{u} \in \mathbf{R}^{N-1}$, and where $\mathbf{E}(ww^T) = W$. We note that this is no longer a trivial Linear Quadratic Regulator (LQR) problem.

In the next chapter, we show how we iteratively can update the feedback matrix $\hat{L}$, such that the cost $J(\hat{L})$ is reduced.

Chapter 6

Modular Distributed Wind Farm Controller

In this chapter we show how an adaptive distributed wind farm controller can be constructed, iteratively lowering the cost of Problem 5.23. We first illustrate the principle of updating the feedback matrix using the gradient descent method. Thereafter we show how the gradient is found distributedly. Finally we show why the gradient descent method is used for controller updates, and what the alternatives are.

6.1 Iterative State Feedback

The basis for the iterative feedback, is to update the optimization variable $\hat{L}$ based on measurements of the states x of the wind farm. Moreover, this is done distributed, such that only local measurements are used when updating the block of $\hat{L}$ associated with each agent.

6.1.1 The Gradient Descent Method

The problem of approximately solving a problem on a form as Problem 5.23 is treated in [MR09], and will be described in the following. The following is based on [MR09] and [MMR11].

The basis of the method, is the gradient descent method. The reason for choosing this algorithm as a basis for the distributed controller, is described in Section 6.4.

We first present the gradient descent method [BV04, p. 466].
repeat

1. Set the descent direction equal to the negative gradient
   $\Delta\hat{L} := -\nabla_{\hat{L}} J(\hat{L})$
2. Choose a step length α by exact or inexact line search
3. Update the feedback matrix
   $\hat{L} := \hat{L} + \alpha\Delta\hat{L}$

until some stopping criterion.

This algorithm must run distributed, such that each agent updates the block of the feedback matrix $\hat{L}$ corresponding to that agent. In the following it will be shown how this can be done by

modifying step 2 in the gradient descent method. As only local information is available at each turbine, neither exact nor inexact line search is an option. Therefore step 2 in gradient descent method is omitted, and a suitable static value of α is used.

This leaves out the problem of finding $\nabla_{\hat{L}} J$ distributed, which is the subject of the following section.

6.2 Distributed Synthesis

In this section, we show how the gradient of Problem 5.23 can be found by solving a set of Lyapunov equations based on the wind farm dynamics. Hereafter we show how this can be done in a distributed manner based on the adjoint system.

6.2.1 Centralized Cost Function Gradient

We observe Problem 5.23. Ignoring the structural constraint on the feedback matrix $\hat{L}$, the problem is given by

$$\begin{array}{ll} \text{minimize} & J(\hat{L}) = \mathbf{E}(x^T Q x + 2x^T \hat{S}\hat{u} + \hat{u}^T \hat{R}\hat{u}) \\ \text{subject to} & x(k+1) = Ax(k) + \hat{B}\hat{u}(k) + B_d w(k) \\ & \hat{u}(k) = -\hat{L}x(k). \end{array} \tag{6.1}$$

For the function $J(\hat{L})$ in Problem 6.1, [MMR11] shows that the gradient is given by

$$\nabla_{\hat{L}} J = 2\left[\hat{R}\hat{L} - \hat{S}^T - \hat{B}^T P(A - \hat{B}\hat{L})\right] X \tag{6.2}$$

where P and X are the solutions to the Lyapunov equations

$$X = A_{\hat{L}} X A_{\hat{L}}^T + W \tag{6.3}$$
$$P = A_{\hat{L}}^T P A_{\hat{L}} + Q_{\hat{L}} \tag{6.4}$$

with the matrices $A_{\hat{L}}$ and $Q_{\hat{L}}$ given by

$$A_{\hat{L}} = A - \hat{B}\hat{L} \tag{6.5}$$
$$Q_{\hat{L}} = Q - \hat{S}\hat{L} - (\hat{S}\hat{L})^T + \hat{L}^T \hat{R}\hat{L} \tag{6.6}$$

and where $W = \mathbf{E}(ww^T)$.

In the following, an argument for Equation 6.2 will be presented. We will do this by rewriting the expression of the cost $J(\hat{L})$, and taking the derivative.

The cost $J(\hat{L})$ can be expressed as

$$\begin{aligned} J(\hat{L}) &= \mathbf{E}(x^T Q x + 2x^T \hat{S}\hat{u} + \hat{u}^T \hat{R}\hat{u}) \\ &= \mathbf{Tr}(\mathbf{E}(x^T Q x + 2x^T \hat{S}\hat{u} + \hat{u}^T \hat{R}\hat{u})) \\ &= \mathbf{Tr}(\mathbf{E}(x^T Q x - 2x^T \hat{S}\hat{L}x + x^T \hat{L}^T \hat{R}\hat{L}x)) \\ &= \mathbf{Tr}(\mathbf{E}(xx^T (Q - 2\hat{S}\hat{L} + \hat{L}^T \hat{R}\hat{L}))) \\ &= \mathbf{Tr}(X(P - A_{\hat{L}}^T P A_{\hat{L}})) \\ &= \mathbf{Tr}(XP - PA_{\hat{L}} X A_{\hat{L}}^T)) \\ &= \mathbf{Tr}(XP - P(X - W)) \\ &= \mathbf{Tr}(PW). \end{aligned}$$

Here it is used, that trace is invariant under cyclic permutations. Also it is used that $\mathbf{E}(xx^T) = X$. This is realized as the system in Problem 5.13 is stationary, which means that we can write

$$\begin{aligned} \mathbf{E}(x(k)x(k)^T) &= \mathbf{E}(x(k+1)x(k+1)^T) \\ &= \mathbf{E}\left[(Ax(k) + \hat{B}\hat{u}(k) + w(k))(Ax(k) + \hat{B}\hat{u}(k) + w(k))^T\right] \\ &= A_{\hat{L}} X A_{\hat{L}}^T + W \\ &= X. \end{aligned}$$

Taking the derivative of the cost function $J(\hat{L})$ can thus be done by taking the derivative of $\mathbf{Tr}(PW)$. First we differentiate the Equation 6.4

$$\begin{aligned} dP &= A^T dP A - A^T dP \hat{B}\hat{L} - A^T P \hat{B} d\hat{L} - d\hat{L}^T \hat{B}^T P A - \hat{L}^T \hat{B}^T dP A + d\hat{L}^T \hat{B}^T P \hat{B}\hat{L} + \\ &\quad \hat{L}^T \hat{B}^T dP \hat{B}\hat{L} + \hat{L}^T \hat{B}^T P \hat{B} d\hat{L} - \hat{S} d\hat{L} - d\hat{L}^T \hat{S}^T + d\hat{L}\hat{R}\hat{L} + \hat{L}\hat{R} d\hat{L} \\ &= A_{\hat{L}}^T dP A_{\hat{L}} + M + M^T \end{aligned}$$

where $M = d\hat{L}^T(\hat{R}\hat{L} - \hat{S}^T - \hat{B}^T P A_{\hat{L}})$. We can write this as

$$dP = (A_{\hat{L}}^T)^i dP A_{\hat{L}}^i + \sum_{k=0}^{i-1} (A_{\hat{L}}^T)^k (M + M^T) A_{\hat{L}}^k, \qquad i \in \mathbf{Z}_+$$

and thus also

$$dP = \sum_{k=0}^{\infty} (A_{\hat{L}}^T)^k (M + M^T) A_{\hat{L}}^k$$

as $A_{\hat{L}}$ is stable.

Now we can get an expression for the desired derivative of the cost

$$\begin{aligned} dJ &= \mathbf{Tr}(dPW) \\ &= \mathbf{Tr}\left(\sum_{k=0}^{\infty} (A_{\hat{L}}^T)^k (M + M^T) A_{\hat{L}}^k W\right) \\ &= 2\,\mathbf{Tr}\left(d\hat{L}^T(\hat{R}\hat{L} - \hat{S}^T - \hat{B}^T P A_{\hat{L}}) X\right) \end{aligned} \tag{6.7}$$

The last equation sign follows as

$$X = \sum_{k=0}^{\infty} A_{\hat{L}}^k W (A_{\hat{L}}^T)^k. \tag{6.8}$$

That Equation 6.8 holds can be realized, as the right hand side is a solution to the Lyapunov Equation 6.3

$$\begin{aligned}\sum_{k=0}^{\infty} A_{\hat{L}}^k W (A_{\hat{L}}^T)^k &= A_{\hat{L}} \sum_{k=0}^{\infty} A_{\hat{L}}^k W (A_{\hat{L}}^T)^k A_{\hat{L}}^T + W \\ &= \sum_{k=1}^{\infty} A_{\hat{L}}^k W (A_{\hat{L}}^T)^k + W \\ &= \sum_{k=0}^{\infty} A_{\hat{L}}^k W (A_{\hat{L}}^T)^k.\end{aligned}$$

Using the rule that $dZ = \mathbf{Tr}(dX^T Y) \Rightarrow \nabla_X Z = Y$ [MMR11], we have the end result by Equation 6.7, that

$$\nabla_{\hat{L}} J = 2[\hat{R}\hat{L} - \hat{S}^T - \hat{B}^T P A_{\hat{L}}]X. \tag{6.9}$$

6.2.2 Distributed Cost Function Gradient

In the distributed control method, the gradient $\nabla_{\hat{L}} J$ must be calculated using only local and neighboring state knowledge, which calls for a reformulation of Equation 6.9. This reformulation is based on the closed loop state equation

$$x(k+1) = (A - \hat{B}\hat{L})x(k) + w(k) \tag{6.10}$$

and on introducing the adjoint state equation with the adjoint states λ defined by

$$\lambda(k-1) = (A - \hat{B}\hat{L})^T \lambda(k) - (Q - \hat{S}\hat{L} - (\hat{S}\hat{L})^T + \hat{L}^T \hat{R}\hat{L})x(k). \tag{6.11}$$

By realizing that

$$\mathbf{E}\, x(k)x(k)^T = X \tag{6.12}$$

$$\mathbf{E}\, \lambda(k)x(k)^T = -P(A - \hat{B}\hat{L})X, \tag{6.13}$$

we can reformulate Equation 6.9 into [MMR11], [MR09]

$$\nabla_{\hat{L}} J = 2((\hat{R}\hat{L} - \hat{S}^T)\, \mathbf{E}\, xx^T + \hat{B}\, \mathbf{E}\, \lambda x^T). \tag{6.14}$$

This reformulation is desired, as it enables us to find the gradient locally. The key is that by local and neighboring measurements of x and simulations of λ, it is possible to locally estimate the blocks of $\mathbf{E}\, x(k)x(k)^T$ and $\mathbf{E}\, \lambda(k)x(k)^T$ relevant to the given agent. With these estimates, it is by 6.14 possible to get an estimate of the block of $\nabla_{\hat{L}} J$ that is relevant to the given agent.

In the following we present a proof of 6.14. It is already previously showed that Equation 6.12 holds. That Equation 6.13 holds can be realized by rewriting the state Equation 6.10 as

$$x(k) = A_{\hat{L}}^i x(k-i) + \sum_{l=1}^{i} A_{\hat{L}}^{l-1} w(k-l), \qquad i \in \mathbf{Z}_+. \tag{6.15}$$

Similarly, we can rewrite Equation 6.11 as

$$\begin{aligned}
\lambda(k) &= A_{\hat{L}}^T \lambda(k+1) - Q_{\hat{L}} x(k+1) \\
&= (A_{\hat{L}}^T)^i \lambda(k+i) - \sum_{j=0}^{i-1} (A_{\hat{L}}^T)^j Q_{\hat{L}} x(k+1+j), \qquad i \in \mathbf{Z}_+ \\
&= -\sum_{j=0}^{\infty} (A_{\hat{L}}^T)^j Q_{\hat{L}} x(k+1+j) \\
&= -\sum_{j=0}^{\infty} (A_{\hat{L}}^T)^j Q_{\hat{L}} \left(A_{\hat{L}}^{j+1} x(k) + \sum_{l=1}^{j+1} A_{\hat{L}}^{l-1} w(k-l+1+j) \right). \qquad (6.16)
\end{aligned}$$

The last equality sign holds as

$$x(k+1+j) = A_{\hat{L}}^{1+j} x(k) + \sum_{l=1}^{1+j} A_{\hat{L}}^{l-1} w(k+1+j-l) \qquad (6.17)$$

by Equation 6.15.

Taking the expected value of $\lambda(k)x(k)^T$ using the expression for $\lambda(k)$ presented in Equation 6.16 yields

$$\begin{aligned}
\mathbf{E}\, \lambda(k)x(k)^T &= \mathbf{E} \left(-\sum_{j=0}^{\infty} (A_{\hat{L}}^T)^j Q_{\hat{L}} \left(A_{\hat{L}}^{j+1} x(k) + \sum_{l=1}^{j+1} A_{\hat{L}}^{l-1} w(k-l+1+j) \right) x(k)^T \right) \\
&= -P A_{\hat{L}} X
\end{aligned}$$

which is the basis of 6.14. Here it is used, that the white noise w is uncorrelated with the state x. Further it is used, that

$$\sum_{j=0}^{\infty} (A_{\hat{L}}^T)^j Q_{\hat{L}} A_{\hat{L}}^j = P \qquad (6.18)$$

which follows from 6.4 as P is stable.

Motivation for the Adjoint System

In the following, we will present a short motivation for finding the adjoint system 6.11, by examining the original minimization problem, and introducing the dual variable.

The optimal value of the original problem, Problem 5.13, is given by $J^\star = J(\hat{L}^\star)$. By neglecting the noise, we can reformulate the problem as

$$\begin{array}{ll}
\text{minimize} & J(x, \hat{u}) = x^T Q x + 2x^T \hat{S} \hat{u} + \hat{u}^T \hat{R} \hat{u} \\
\text{subject to} & x(k+1) = Ax(k) + \hat{B}\hat{u}(k)
\end{array} \qquad (6.19)$$

with variables x and $\hat{u}$. If we take the Lagrangian associated with Problem 6.19, we get [MR10]

$$\Lambda(x, \hat{u}, \lambda) = x^T Q x + 2x^T \hat{S} \hat{u} + \hat{u}^T \hat{R} \hat{u} + 2\lambda^T (x(k+1) - Ax(k) + \hat{B}\hat{u}(k)). \qquad (6.20)$$

Based on this, we can find the Lagrange dual function, as the minimum value of the Lagrangian Λ over $x, \hat{u}$ [BV04, p. 216]. Since $\Lambda(x, \hat{u}, \lambda)$ is a convex quadratic function of $x, \hat{u}$, we can find

the minimizing $x, \hat{u}$ by [BV04, p. 219], [MR10]

$$\begin{aligned} 0 &= \nabla_x \Lambda(x, \hat{u}, \lambda) = 2(Qx(k) + \hat{S}\hat{u}(k) + \lambda(k-1) - A^T\lambda(k)) \\ 0 &= \nabla_u \Lambda(x, \hat{u}, \lambda) = 2(\hat{S}^T x(k) + \hat{R}\hat{u}(k) - \hat{B}^T\lambda(k)) \end{aligned}$$

and by inserting $\hat{u} = \hat{L}x$

$$\begin{aligned} 0 &= \nabla_x \Lambda(x, \lambda) = 2(Qx(k) - \hat{S}\hat{L}x(k) + \lambda(k-1) - A^T\lambda(k)) \\ 0 &= \nabla_u \Lambda(x, \lambda) = 2(\hat{S}^T x(k) - \hat{R}\hat{L}x(k) - \hat{B}^T\lambda(k)). \end{aligned}$$

This allows us to write $0 = \nabla_x \Lambda(x, \lambda) - \hat{L}^T \nabla_u \Lambda(x, \lambda)$ which leads to

$$\lambda(k-1) = (A - \hat{B}\hat{L})^T\lambda(k) - (Q - \hat{S}\hat{L} - (\hat{S}\hat{L})^T + \hat{L}^T\hat{R}\hat{L})x(k) \tag{6.21}$$

which is identical to Equation 6.11.

Another motivation is, that we know that distributed control is tightly related to the adjoint systems [Ran09], [MR10], [Boy10]. Also [GL00, p. 443] suggests a similar use of the adjoint system, as an approach to iteratively finding the solution to a dynamic optimization problem.

6.2.3 Distributed System Simulation

Based on the above, we will show how the feedback matrix $\hat{L}$ can be updated iteratively in a distributed manner. Each agent updates the block of the feedback matrix relevant to the given agent based on own measurements, and measurements from neighboring agents.

Two things are required for this distributed control to be possible. It must be possible to simulate the adjoint states $\lambda(k)$ locally on each agent. Also, it must be possible to calculate the gradient $\nabla_{\hat{L}} J$ as described by Equation 6.14 locally. Locally means, that it must be possible to do on each individual agent using only own data and data obtained from neighboring agents. In the following we will describe that this indeed is possible, due to the structure of the System 5.13.

Let us first consider the state x and the adjoint state λ. The state x can be measured locally and shared with upwind and downwind turbines.

The adjoint states must on the contrary be simulated in the backward direction based on the measurements according to Equation 6.11. We therefore examine the structure of the terms of Equation 6.11. The blocks in the term $(A - \hat{B}\hat{L})^T$ have the structure such that

$$(A - \hat{B}\hat{L})^T \in \{X \,|(X)_{i,j} \neq 0 \quad \text{only if} \quad i - l_1 \leq j \leq i + l_2 + 1\}$$

where $(X)_{i,j}$ denotes block i, j.

This means, that it is necessary for each turbine to know the adjoint state of l_1 turbines upwind and $l_2 + 1$ turbines downwind.

For the second term in Equation 6.11, we observe that the blocks are structured such that

$$(Q - \hat{S}\hat{L} - (\hat{S}\hat{L})^T + \hat{L}^T\hat{R}\hat{L}) \in \{X \,|(X)_{i,j} \neq 0 \quad \text{only if} \quad i - l_1 - 1 \leq j \leq i + l_2 + 1\}.$$

This means, that to simulate the adjoint states, each turbine need to know the state of $l_1 + 1$ turbines upwind and $l_2 + 1$ turbines downwind.

The above concludes, that communication with $l_1 + 1$ turbines upwind and $l_2 + 1$ turbines downwind, allows each turbine to know its own state and adjoint state. Next we show that each agent is able to calculate the local block of the gradient of the feedback matrix.

6.2.4 Distributed Gradient Calculation

We examine finding the gradient $\nabla_{\hat{L}} J$ distributed, based on local knowledge of the state and adjoint state, using Equation 6.14. This is done by observing the two terms in Equation 6.14. The blocks of the first term are structured such that

$$(\hat{R}\hat{L} - \hat{S}^T) \in \{X \,|(X)_{i,j} \neq 0 \quad \text{only if} \quad i - l_1 - 1 \leq j \leq i + l_2 + 1\}.$$

This means that communication with $l_1 + 1$ upwind turbines, and $l_2 + 1$ turbines downwind is necessary. With this communication enabled, $\mathbf{E}\, xx^T$ can be estimated as

$$(\mathbf{E}\, xx^T)_{\text{est}} = \frac{1}{K} \sum_{t=t_k}^{t_k+K} x(t)x(x)^T \tag{6.22}$$

where K is the number of sample times between each update of the feedback matrix.

The second term in Equation 6.14 is $\hat{B}^T$, which has the structure

$$\hat{B}^T \in \{X \,|(X)_{i,j} \neq 0 \quad \text{only if} \quad i - l_1 \leq j \leq i + l_1 + 1\}.$$

This means that communication with $l_1 + 1$ upwind turbines, and l_2 turbines downwind is necessary. With this communication enabled, $\mathbf{E}\, \lambda x^T$ can be estimated as

$$(\mathbf{E}\, x\lambda^T)_{\text{est}} = \frac{1}{K} \sum_{t=t_k}^{t_k+K} \lambda(t)x(x)^T. \tag{6.23}$$

6.3 Control Algorithm

We can now write the algorithm for the distributed controller. The basis of the algorithm is that each turbine is able to access the states of $l_1 + 1$ upwind turbines and $l_2 + 1$ turbines downwind, and to access the simulated adjoint states of l_1 turbines upwind and $l_2 + 1$ turbines downwind.

The algorithm for turbine i can be written as [MMR11]

repeat

1. Measure local states and obtain states from neighbors, such that $[x_{i-l_1-1}(t), \ldots, x_{i+l_2+1}(t)]$ is known for $t = t_k, \ldots, t_{k+K}$.

2. Simulate adjoint states based on local and neighboring states and adjoint states, such that $[\lambda_{i-l_1}(t), \ldots, \lambda_{i+l_1+1}(t)]$ is known for $t = t_k, \ldots, t_{k+K}$, based on Equation 6.11.

3. Estimate the correlation of the states, and the correlation of the states and the adjoint states, based on $K + 1$ measured and simulated states, using Equations 6.22 and 6.23.

4. Estimate the block of the gradient corresponding to the turbine based on the correlation estimations using Equation 6.14.

5. Update the block of the gradient corresponding to the turbine by letting $\hat{L}_{i,j} := \hat{L}_{i,j} + \alpha \nabla_{\hat{L}} J_{i,j}$, using a static step size α.

6. Let $k = k + K$.

6.4 Alternative Algorithms

In this section, a number of alternative optimization algorithms will be described with focus on the possibility of using them in distributed control. This will also reveal, why the gradient descent method is chosen above.

The reason that it is desired to find an alternative method, is to avoid the use of a static value of α, and to increase performance.

The three alternative algorithms that are considered in the following, all seek to exploit second order properties of the optimization problem, in contrast to the gradient descent only using first order properties. Therefore we perform a second order Taylor approximation $\hat{J}$ of J at $\hat{L}$

$$\hat{J}(\hat{L}+v) = J(\hat{L}) + \nabla J(\hat{L})^T v + \frac{1}{2} v^T \nabla^2 J(\hat{L}) v. \tag{6.24}$$

With this second order model, we capture more of the properties of the function $\hat{J}$, that the first order model used in the gradient descent method. It is therefore expected that these methods will increase the performance of the algorithm.

6.4.1 Newton's Method

In Newton's method, it is used that $\hat{J}(\hat{L}+v)$ is a quadratic convex function in v. We can find the minimum over v as

$$\frac{\partial}{\partial v}\hat{J}(\hat{L}+v) = \nabla J(\hat{L}) + \nabla^2 J(\hat{L}) v = 0 \tag{6.25}$$

which gives

$$v = -\nabla^2 J(\hat{L})^{-1} \nabla J(\hat{L}). \tag{6.26}$$

This is the background for Newton's algorithm, which is [BV04, p. 487]

repeat

1. $\Delta\hat{L} := -\nabla^2 J(\hat{L})^{-1} \nabla J(\hat{L})$
2. $\hat{L} := \hat{L} + \Delta\hat{L}$

until some stopping criterion.

A benefit of this method, is that a step size does not have to be chosen. We can therefore avoid the use of a static α, as desired.

The method requires that the Hessian $\nabla^2 J(\hat{L})$ can be found distributedly. In [GL00, p. 443] it is suggested, that the Hessian also can be found by examination of the adjoint equation. But even if the Hessian can be found in a distributed manner, the inversion of the Hessian would destroy the sparsity structure of the matrix, and make distribution of the algorithm impossible.

The use of Newton's method in the distributed controller is therefore not examined further.

6.4.2 Conjugate Direction Method Using the Hessian

The conjugate direction method is based on the minimization of a quadratic function $f(x) = x^T Q x + b^T x,\ Q \geq 0$. We use the quadratic approximation $\hat{J}$ of J as described in Equation 6.24, and associate Q with $\nabla^2 J(\hat{L})$.

One form of the conjugate direction method algorithm is as follows [Lue84, p. 252].

given some $\hat{L}_0$ initialize with $g_0 := \nabla J(\hat{L}_0)^T$ and $d_0 := -g_0$.

for $k = 1 \dots N$

1. Find the step length as
   $\alpha_k := -g_k^T d_k / d_k^T \nabla^2 J(\hat{L}_k) d_k$.

2. Update the solution in the direction d_k
   $\hat{L}_{k+1} := \hat{L}_k + \alpha_k d_k$.

3. Evaluate the gradient
   $g_{k+1} := \nabla J(\hat{L}_{k+1})^T$.

4. Find a new conjugate descent direction
   $\beta_k := g_{k+1}^T \nabla^2 J(\hat{L}_k) g_{k+1} / d_k^T \nabla^2 J(\hat{L}_k) d_k$
   $d_{k+1} := -g_{k+1} + \beta d_k$.

The basis of the conjugate algorithm is as follows. Instead of always seeking for the solution in the negative descent direction $g_k = -\nabla J(\hat{L}_k)^T$, as in the gradient descent method, it seeks along a direction d_k, where this direction is dependent on all previous search directions $d_i,\ i < k$. The method assures, that the search direction d_k is conjugate to all previous search directions. With conjugate search direction we mean, that for the original minimization problem $f(x) = x^T Q x + b^T x$ it holds that

$$d_k^T Q d_i = 0, \quad \text{for} \quad i < k. \tag{6.27}$$

Hereby searching along directions we have already previously searched along is avoided. In contrast to the gradient descent method, the minimization of a quadratic function will be independent of the condition number of the matrix. In the gradient descent method, the path of descent is often a zig-zag movement, indicating that we essentially searched in the same directions again and again. This is avoided in the gradient descent method, as the algorithm insists that the new direction is conjugate to the previous directions. In the pure quadratic case, we are guaranteed to find the optimum in maximum n steps, where n is the dimension of the matrix Q [Lue84, p. 244].

The reason that β is chosen as described in the algorithm, is that in the quadratic case, this choice makes the method a conjugate direction method. The motivation for choosing α as described, is that this choice minimizes the quadratic approximation of the problem in each step [Lue84, p. 245].

In contrast to Newton's method, it is not needed to invert the Hessian, destroying the sparsity pattern. It is however seen, that in order to determine α and β, full knowledge of the direction d and the gradient g is needed. This renders this method unsuited for distributed control.

6.4.3 Conjugate Gradient Method Using Fletcher-Reeves Method

The conjugate gradient method described above can be rewritten, such that we avoid the direct use of the Hessian to represent Q [Lue84, p. 253]. Again, the method is based on the quadratic minimization problem on the form $f(x) = x^T Q x + b^T x,\ Q \geq 0$.

This rewritten version of the conjugate gradient method differs from the previous algorithm in two ways. The first is, that instead of calculating the α that minimizes the quadratic approximation of the problem, a line search is performed in the conjugate direction d on the original function J. This line search might be exact or inexact, *e.g.* a backtracking line search. In this way, this method resembles the damped Newton's method.

The second change, is that the Fletcher-Reeves method is used for determination of β, assuring that the method is a conjugate gradient method. In the quadratic case, this yields the same value of β, as the previous method, and thus is a conjugate gradient method.

Below, the algorithm is presented [Lue84, p. 253].

given some $\hat{L}_0$ initialize with $g_0 := \nabla J(\hat{L}_0)^T$ and $d_0 := -g_0$.

for $k = 1 \dots N$

1. Find the step length by exact line search
   $\alpha_k := \text{argmin}_{\alpha_k} J(\hat{L}_k + \alpha_k d_k)$
   or use an inexact line search algorithm.

2. Update the solution in the direction d_k
   $\hat{L}_{k+1} := \hat{L}_k + \alpha_k d_k$.

3. Evaluate the gradient
   $g_{k+1} := \nabla J(\hat{L}_{k+1})^T$.

4. Find a new conjugate descent direction using Fletcher-Reeves method
   $\beta_k := g_{k+1}^T g_{k+1}^T / g_k^T g_k^T$
   $d_{k+1} := -g_{k+1} + \beta d_k$.

It is possible to implement this method distributedly, by making a few changes. As in the gradient descent method, the choice of the step length α can not be done in a distributed manner, as this requires knowledge of the whole system. We therefore as in the gradient descent method, have to use a static value of α.

The calculation of β requires full knowledge of the gradient g, in order to make the search directions d_i conjugate. This full knowledge is not available. This means that we can not require the full search directions d_i to be conjugated to each other. However, we can insist that for each agent, the local search direction is conjugate to the previous search directions. This idea will be explained in the following.

The variable is the feedback matrix $\hat{L} \in \mathbf{R}^{N-1 \times Np}$. The descent directions d_i will thus have the same structure

$$d_i = \begin{bmatrix} d_i^1 \\ \vdots \\ d_i^{N-1} \end{bmatrix} \tag{6.28}$$

with $d_i^k \in \mathbf{R}^{1 \times Np}$. The original conjugate gradient method, would require the entire search direction d_i to be conjugate to the previous search directions $d_j,\ j < i$. In the distributed control we modify this requirement so that the conjugate direction requirement is on the basis of each agent. This means that d_i^1 is conjugated to $d_j^1,\ j < i$ that d_i^2 is conjugated to $d_j^2,\ j < i$ etc.

The interpretation is, that each local search direction is approximately conjugate to the previous local search directions.

6.4.4 Choice of Optimization Algorithm

Based on the above, one candidate is found as an alternative to the gradient descent method, namely the conjugate gradient method using the Fletcher-Reeves method. This method does not resolve the problem of finding an appropriate value of the step size α. But the hope is that this searching algorithm will converge faster, due to the fact that the method insists on making each agent's search direction conjugate on each other.

The two algorithms are applied to a system similar to the one presented in [MR09]. Here 10 agents are present, located in one line, each with one input and one output. Each agent can communicate with the closest neighbor on each side. Further details are found in [MR09] or in the script.

The code utilizing the distributed gradient descent method, is provided by the authors of [MR09], and edited to include the conjugate gradient method. It can be found in:

`matlab/optimization_algorithms/optimization_alg_comparison.m`

Two simulations are conducted, one where $(\mathbf{E}\, xx^T)_{\text{est}}$ and $(\mathbf{E}\, x\lambda^T)_{\text{est}}$ are based on 1000 measurements, and one where they are based on 20 measurements. In both cases, a static value of α is used. The convergence of the solution is shown in Figure 6.1, with the long simulation time in the left figure, and the short simulation time in the right figure.

The figures show to interesting results. The figure to the left shows a faster convergence, for the conjugate gradient method, compared to the gradient descent. Similar results are obtained applying different noise sequences. This illustrates, that the conjugate gradient method indeed converges faster.

The figure to the right, with the short simulation time however shows a different result. Here the conjugated gradient method performs worse than the gradient descent method. Repeating the simulation reveals similar or even worse performance of the conjugate gradient method.

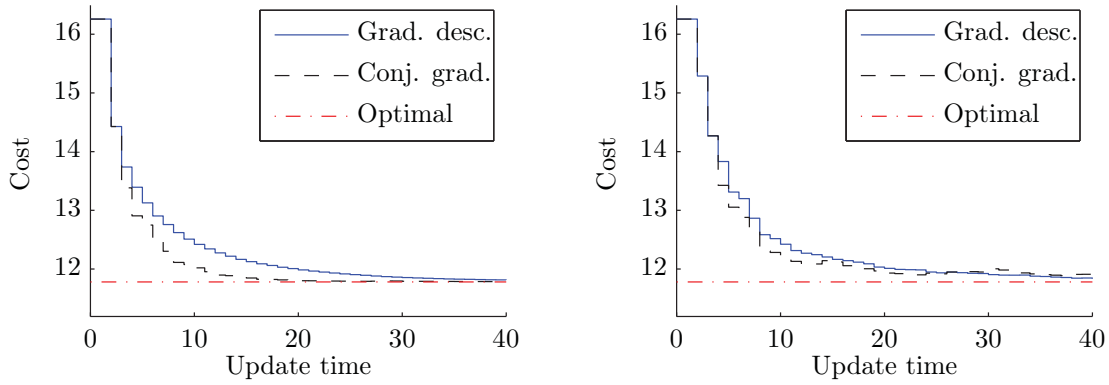

Figure 6.1: Comparison between the gradient descent and the conjugated gradient method with a large number of measurements for each update (left) and a small number of measurements for each update (right).

Based on these simulations, we conclude that the conjugated gradient method requires data with very little noise. Therefore it is chosen not to use the conjugate gradient method, but rather the gradient descent method. The reason is that the conjugate gradient method does not resolve the problem of using a static value of α, and that the method is more sensitive to noise.

In the next chapter, we show how the controller derived in this chapter performs in a realistic simulation environment.

Chapter 7

Controller Performance Evaluation

This chapter describes how the performance and functionality of the modular distributed wind farm controller is evaluated, first on the linearized system, then on the more realistic wind farm environment, using models of National Renewable Energy Laboratory (NREL) wind turbines.

7.1 Evaluation in Linear Wind Farm

In this section, the performance of the adaptive distributed control algorithm is evaluated on the linearized wind farm using a quadratic cost function.

7.1.1 Convergence

We apply the algorithm described in Section 6.3 to a wind farm with $N = 10$ wind turbines. It is assumed that the turbines are placed in one row. Each wind turbine model is linearized in the operating point $\overline{v^i} = 15$ m/s, $\overline{P^i_{\text{dem}}} = 3$ MW. A wind field generated according to the wind spectrum discussed in Section 4.3 on page 22 with a standard deviation of 1.5 is used as exogenous input. An example of this wind field is seen in Figure 7.3 (right). It is further assumed, that the total power demand to the whole wind farm, consisting of 10 turbines, is 30 MW.

Two different communication paths are used. The first is $l_1 = l_2 = 0$, where the turbines are allowed to communicate with one upwind and one downwind turbine. Secondly $l_1 = l_2 = 1$, where the turbines are allowed to have communication with two upwind and two downwind turbines.

Appropriate penalization of the input signals and output signals are chosen, through the Y and U matrices, see Section 5.4. An appropriate static value of α is further chosen, so that the algorithm can run distributedly

We evaluate the algorithm by allowing the system to sample the states 100 times, then update the feedback matrix $\hat{L}$ based on the measurement, and based on simulated adjoint states. By letting the controller update the feedback matrix 250 times, the cost J depicted in Figure 7.1 is obtained. The cost is evaluated as $\mathbf{Tr}(PW)$, where P depends on L, as previously explained. The feedback matrix is initialized as $L = 0$, corresponding to no feedback.

The sample time used in the following results is 10 Hz. Similar results are obtained with sampling rates as slow as 1 Hz.

The conversion shows that both communication patterns lowers the cost J compared to having no feedback, $\hat{L} = 0$. It also shows, that it is beneficial to use the communication pattern $l_1 = l_2 = 1$, *i.e.* allowing communication with two upwind and two downwind turbines. At the same time it

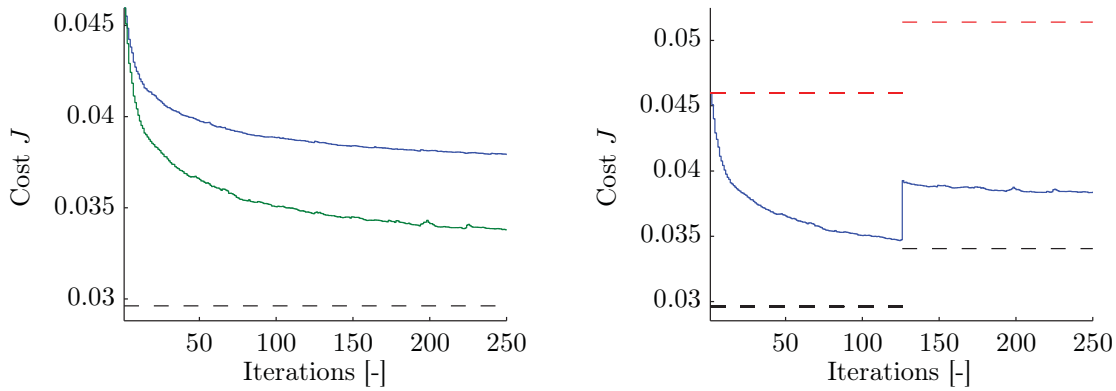

Figure 7.1: Convergence plots of the cost J. The left figure illustrates the use of two different communication patterns, $l_1 = l_2 = 0$ (blue) and $l_1 = l_2 = 1$ (green), compared to optimal feedback (black dashed). The right figure illustrates the convergence, as the wind field changes after 125 iterations. The fatigue with optimal feedback (black dashed) and no feedback (red dashed) are shown as references.

is observed, that increasing the number of wind turbines, that are allowed to communicate, does not increase the performance significantly. The communication pattern $l_1 = l_2 = 1$ is therefore chosen.

In Figure 7.1 (right), the same controller is applied to a wind farm in same operation point. After 125 iterations the wind field is changed, from 15 m/s to 14 m/s, changing the dynamics of the wind turbine. Again the cost J of the adaptive controller is compared to the optimal feedback controller with full state feedback (black dashed). Also it is compared to no feedback, $\hat{L} = 0$ (red dashed).

The plot shows two interesting results. The first is, that the iterative controller found in the wind field 15 m/s has significant better performance than no feedback in the perturbed wind field. Secondly we see, that the controller slightly adapts to the new wind field.

Note here, that the adaption arises due to the estimation of the correlation of states as obtained by system state measurements. The adjoin state is however still based on the original system.

Finally note, that the wind farm controller is developed based on the operating point $\overline{v^i} = 15$ m/s, but still increases performance when $v^i = 14$ m/s. This illustrates that the controller is insensitive to perturbations, which is desired.

The script is found in

`matlab/thanet_linear/thanet.m`

7.1.2 Fatigue Reduction

In the simulation described above, we now observe how the adaptive distributed controller affects the measures of fatigue, that we want minimize. The cost J shown above, is a combination of fatigue due to shaft torsion M_s, tower torsion M_t and use of control signal. In the following we isolate and evaluate the different terms.

Figure 7.2 (left) shows the normalized two norm of the torque on the tower, as the control law iterates. For simplicity, the scale is normalized, so that the two norm of the torque on the tower with no feedback corresponds to 1. Similarly, Figure 7.2 (right) shows the convergence of the

normalized two norm of the torque on the shaft.

Both curves show the desired behavior, that the torques are reduced as the feedback law is updated. We note that the torque on the shaft is reduced with more than 50 %, while the tower torque is reduced more than 10 %.

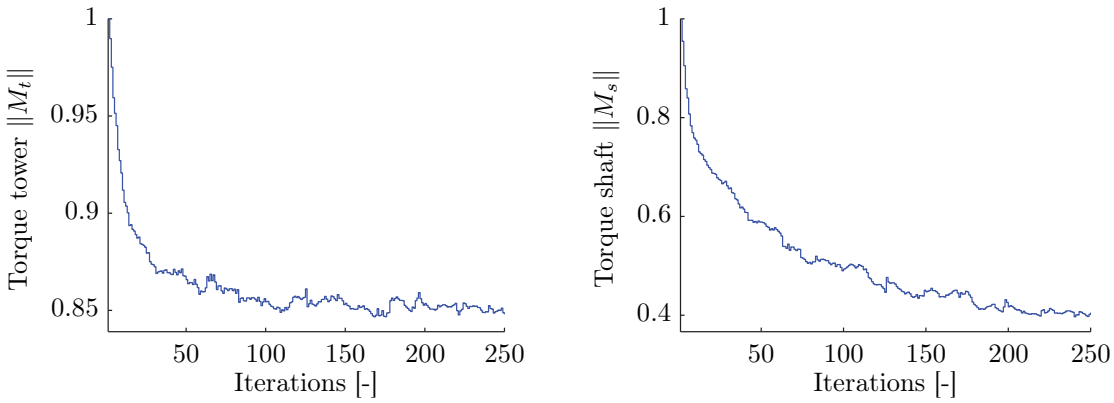

Figure 7.2: Convergence plots of the normalized tower fatigue M_t (left) and shaft fatigue M_s (right).

In those simulations, it is observed that the objective of minimizing M_s and minimizing M_t are conflicting objectives. If solely the torque on the tower is minimized, the torque on the shaft will increase, and vice versa. The chosen trade-off parameter however lowers both objectives.

7.1.3 Transient Responses

We look at some transient responses, where the farm loop is closed using the final controller found after 250 iterations.

Figure Figure 7.3 (right) shows the applied wind field.

Figure 7.3 (left) shows the change in power demand P^i_{dem} for the turbines $i = 1, 2, 3$. This figure illustrates how the power set-points of the turbines in the farm are dynamically updated. It is observed that the perturbations are in the magnitude of 0.1 MW, which we compare to a set-point of $P^i_{\text{dem}} = 3$ MW, illustrating that no saturation is violated.

Also we observe the outputs we want to minimize, namely the tower and shaft torques. We compare the torques on tower and shaft with and without wind farm control. This is presented in Figure 7.4 for one of the turbines. Here we observe by inspection that the shaft torque M_s is reduced (right), while a slight reduction in tower torque M_t can be seen.

7.1.4 Performance in Various Conditions

In this section we evaluate the performance of the wind farm controller in various conditions. This is done by evaluating the controller on the linear system, linearized in three different wind fields, and three different power set-points. All operating points are chosen, so that the wind turbines are in the power tracking region. Again 10 wind turbines are used with communication pattern $l_1 = l_2 = 1$ and a wind variation as before.

Table 7.1 shows the results. In each cell we show the reduction of the two norm of both the tower and the shaft torques. The reduction in percent of the tower torque is shown in the upper part of

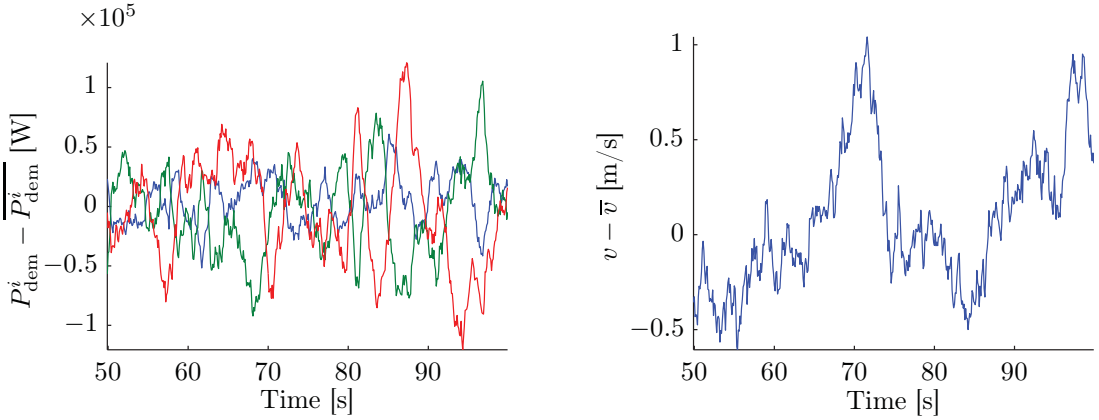

Figure 7.3: Transient responses after applying the iterative controller. The left figure shows the variation in power demand for three of the turbines in the wind park. The right figure illustrates the wild field.

the cell, while the reduction in percent of the shaft torque is shown in the lower part. The scrip is found in

`matlab/thanet_linear/thanet_table.m`

The results show that reductions of the torque on the tower in the magnitude of 12 % can be expected, while reductions of the torque on the shaft in the magnitude of 60 % can be expected.

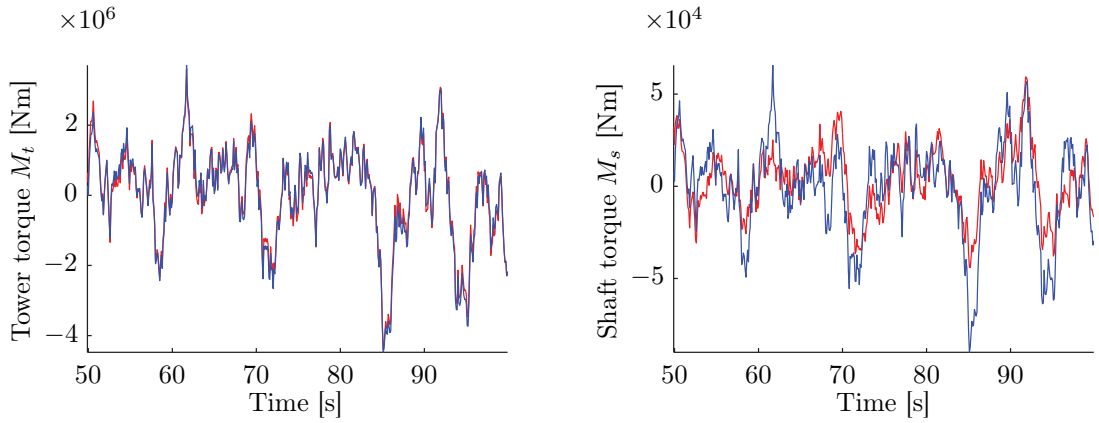

Figure 7.4: Tower and shaft torques with wind farm control (red) and without wind farm control (blue)

7.2 Evaluation in NREL Wind Farm

To make a more realistic evaluation of the distributed wind farm controller, we use a wind farm simulation environment consisting of 10 NREL wind turbines. We use the models of the NREL wind turbines as described in [JBMS09] and as implemented in [Aeo10b]. Again the total power demand to the wind farm is 30 MW. The controller and wind field presented in last section is used.

Fatigue reduction [%]	$\overline{v} = 15$ m/s	$\overline{v} = 20$ m/s	$\overline{v} = 25$ m/s
$P_{\text{dem}} = 2.0$ MW	13/62	9/65	8/60
$P_{\text{dem}} = 3.0$ MW	15/61	11/65	9/63
$P_{\text{dem}} = 4.0$ MW	18/63	13/67	10/65

Table 7.1: Each cell shows the reduction of $\|M_t\|$ followed by the reduction of $\|M_s\|$ in %.

7.2.1 Convergence

The communication pattern $l_1 = l_2 = 1$ is used which means that the turbines are allowed to communicate with two upwind and two downwind turbines. As before, appropriate penalization matrices Y, U are chosen as the basis of the controller design.

A full order observer observes the wind states that are not measurable, and the estimated states are used by the wind farm controller.

The feedback matrix is updated every 100 samples. Figure 7.5 shows the convergence over 400 such iterations, illustrating that the wind turbine controller indeed converges in this NREL based wind farm. Note that the convergence time is longer, than in the linear case. The reason is, that a smaller step size α must be chosen in order to assure convergence.

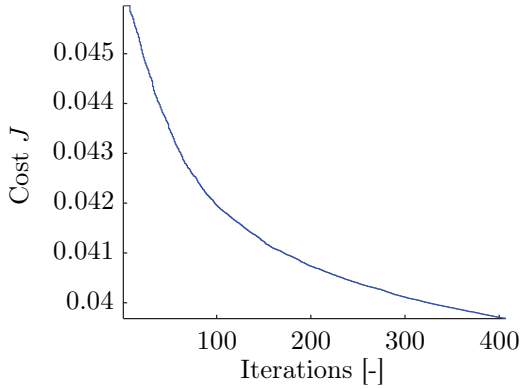

Figure 7.5: Convergence of the feedback feedback matrix when applying the controller to a wind farm model consisting of 10 NREL wind turbines.

Figure 7.5 illustrates, that the controller also converges in this NREL wind farm environment.

Note, however, that the cost J plotted in Figure 7.5 is based on the linearized system. This means that the plot does not illustrate the true cost, but an approximation of the cost evaluated as $\mathbf{Tr}(PW)$. The reason that this is only an approximation of the cost, is that the matrix P is determined based on the linearized system, which is an approximation of the NREL wind turbine. Later simulations will show the actual performance.

As it requires extensive simulations to evaluate the actual cost of the system depending on the feedback matrix, we omit making a plot similar to last section, where we evaluate the performance after a change of wind field.

The scrips and models for the above simulation is found in

`matlab/thanet/thanet.m`

In the above simulations, the sample time used is 10 Hz is used. Similar results are obtained with sampling rates as slow as 1 Hz.

7.2.2 Transient Responses

We now look at some transient responses for the wind turbine farm, as obtained using the final controller of the previous simulation, *i.e.* the controller found after 400 iterations.

Figure 7.6 (left) shows the change in power demand P^i_{dem} for the 10 turbines in the wind farm as changing over 10 seconds. We see that the power reference change is within the desired region, as they each are kept around the operating point of 3 MW, and certainly within the possible region of power production for the turbines. We can also note, that they contain no high frequency components, which is the reason that similar performance can be achieved at slower sampling rates.

The wind field used, is the same as previously used, see Figure 7.3.

An interesting observation that can be made in this simulation, is the evaluation of the actual power production of the wind farm P_{out}. As described in the modeling section, the linear transfer function from power demand and power output is assumed unity. Therefore we could not perform this evaluation in the linear model based simulation in last section.

We compare the total power output P_{out} of the wind farm with and without the controller. This is illustrated in Figure 7.6 (right) over 10 s. The response illustrates, that the power output on farm level is kept close to the total set-point of 30 MW. Further, almost no difference is seen in the case with controller compared to the case without control.

Last we compare the signals we want to control, namely the position of the nacelle p_{nac} and the torsion of the shaft θ. This is illustrated in Figure 7.7. Again a 10 s sample is shown for one of the 10 turbines. By inspection we again notice a slight reduction in the tower torque, and a more significant reduction in the shaft torque.

Note here, that these plots of course do not capture the total effect of the controller, as just 10 s of simulation is illustrated for only one of the 10 turbines. The plots are, however, representative for the behavior of the controller, as is evident from the next section.

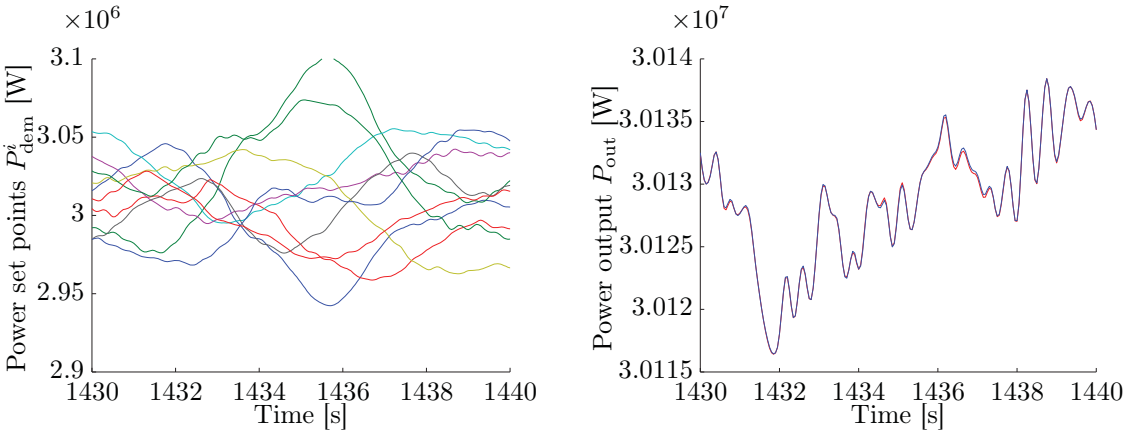

Figure 7.6: The left figure shows the power set-points for the 10 turbines in the NREL wind farm over 10 s. The right figure shows the farm power production using the controller found by iteration (red), compared to the farm power production with no control (blue).

The script is found in

`matlab/thanet/thanet_fatigue_evaluation.m`

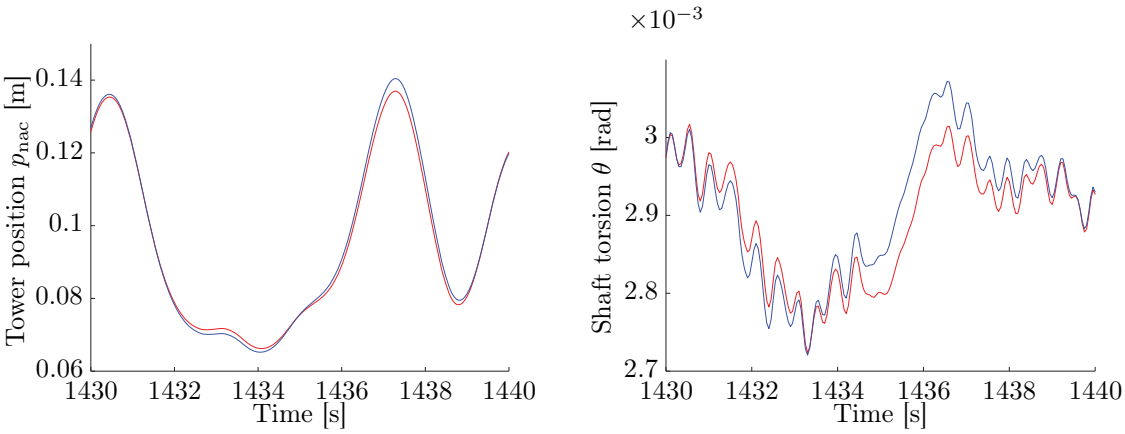

Figure 7.7: The left figure shows the sway of the wind turbine tower while the right figure shows the torsion of the generator shaft. In both plots we compare the controller found by iteration (red) with open loop control (blue).

Conditions	Reduction $\mathcal{Q}(M_t)$	Reduction $\mathcal{Q}(M_s)$	Reduction $\mathcal{R}(p_{\text{nac}})$	$\mathcal{R}(\theta)$
$P_{\text{dem}} = 30$ MW, $\overline{v} = 15$ m/s	7 %	21 %	10 %	53 %
$P_{\text{dem}} = 30$ MW, $\overline{v} = 20$ m/s	4 %	17 %	7 %	42 %

Table 7.2: Fatigue reduction measured using a quadratic function and using a rainflow counting algorithm in different operation regions of the wind turbines.

7.2.3 Performance in Various Conditions

To evaluate the actual performance of the system, simulations of 4000 s are conducted. Based on these simulations, the performance can be found by evaluating rainflow counting on the position of the nacelle and rainflow counting on the shaft, revealing the fatigue on the tower and fatigue on the shaft.

The simulations are conducted with the communication pattern $l_1 = l_2 = 1$ for 10 wind turbines. The controller is found by letting the system update the controller 400 times, where each controller update is based on 100 measurements. In the performance evaluation, the controller is kept constant.

The results are presented in Table 7.2, where we compare the performance with and without wind farm control. We observe that both tower fatigue and shaft fatigue is reduced.

We note that the fatigue is reduced most in the rainflow counting sense. This is a good result, as this is the value we consider closest to actual fatigue. The result is not surprising, as the earlier relationships between the two measures of fatigue illustrate that a small reduction of $\mathcal{Q}(M_s)$ and $\mathcal{Q}(M_t)$, will generate large decrease in $\mathcal{R}(\theta)$ and $\mathcal{R}(p_{\text{nac}})$, see Figures 5.5 and 5.6.

The final conclusion of this first part of the Thesis is presented in Chapter 12.

Part II

Distributed Controller for Thanet Wind Farm

Chapter 8

Outline Part II

This chapter describes the outline of this second part of the Thesis. In this part of the work, a simplified controller is developed suitable for implementation at real wind turbines. It is planned, that this wind farm control will be evaluated in a real wind farm experiment at Thanet wind farm in the summer of 2011.

- **Thanet Wind Farm Experiment Description** , Chapter 9
  This chapter describes possibilities and limitations for the upcoming feedback experiments at Thanet wind farm.
- **Control Strategy**, Chapter 10
  This chapter describes how a simplified wind farm controller suitable for implementation at Thanet wind farm can be designed. The goal of the controller is to illustrate the concept of wind farm control.
- **Controller Design and Evaluation**, Chapter 11
  This final chapter describes how to design the wind farm controller, accommodating limitations of the wind turbines in the Thanet wind farm. Further, the controller is evaluated on a simulation model of a wind farm.

Chapter 9

Thanet Wind Farm Experiment Description

This chapter describes wind turbine experiments planned to be performed at Thanet wind farm in the summer 2011, as a part of the Aeolus research project. Both the wind farm, the turbines in the farm and the wind farm server are described, providing an understanding of the possibilities and limitations of an implementation of wind farm control.

9.1 Experiment Goal

The goal of the feedback control experiment at Thanet wind farm is to illustrate than an increase in performance is achievable letting the wind turbines in the wind farm vary their power set-points dynamically. Demonstrating this in a real wind farm is a strong argument for further research in this area.

9.2 Experiment Setup

The Aeolus research project has since 2009 worked with the understanding of the coupling of wind turbines in large scale wind farms. As a primary source for development of both wind farm models as well as controllers, a number of experiments at the offshore wind farm Thanet are to be performed [Ves11].

Thanet wind farm is located east of Thanet in the UK, and consists of 100 Vestas 3 MW wind turbines [Ofg09], each located with a separation of 500 meters between each turbine, and 800 meters between each row, see Figure 9.1.

In the Aeolus project, it is planned to evaluate a number of tests on Thanet wind farm. These experiments include [Aeo11]

- Validation of single V90 turbine model.
- Validation of static and dynamic models relating effective wind speed at downwind turbines with wind speed and the thrust coefficient of upwind turbines.
- Evaluation of static feedforward control.
- Evaluation of feedback control.

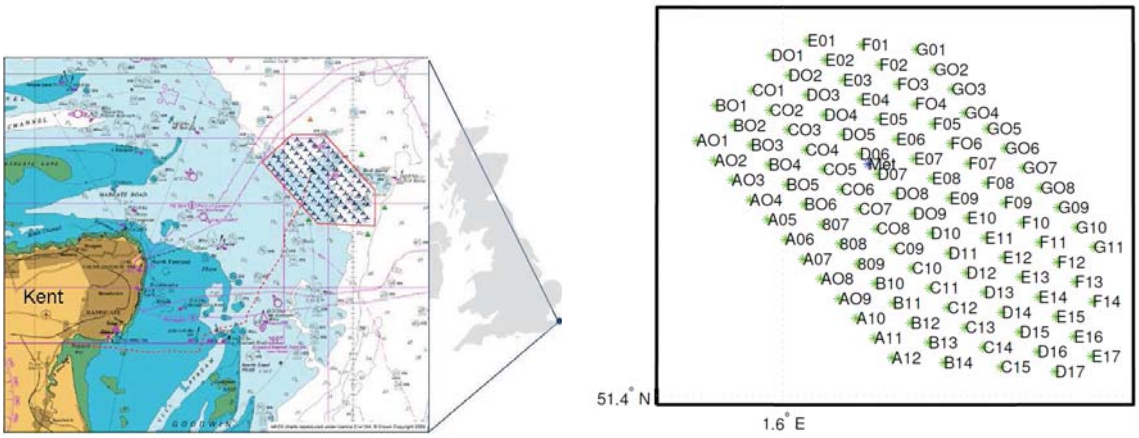

Figure 9.1: Thanet wind farm, located 11 km off the cost of Thanet (left) [Ofg09]. The 10 wind turbines selected for experiments are `A01-A08` and `B03-B04` (right) [Ves11].

This work treats the last item, namely feedback control.

In order to evaluate the above four tests, it has been chosen to use the 10 wind turbines `A01-A08` and turbines `B03-B04`, see figure 9.1. This choice is made such that 8 wind turbines are in a row, experiencing the wake of upwind turbines when the wind direction is along the row. Also two wind turbines are located in the ambient, and thus unaffected by the wake of the 8 turbines, serving the role as a reference [Ves11].

9.3 System Limitations

The turbines in the wind farm are Vestas V90 3 MW wind turbines monitored and controlled by a Supervisory Control And Data Acquisition (SCADA) server system. The performance of this system, determines to what extend the experiments can be executed. The current server system is not build with dynamic wind farm feedback control in mind, and therefore only offers limited possibilities options for this type of control.

In the following we present the possibilities and limitations of the SCADA server system.

Measurements of Signals

It is possible for the SCADA system controlling the wind farm to log 200 - 250 signals with a frequency of 1 Hz. It is chosen to measure the following signals during the experiments.

- Generator angular velocity
- Nacelle direction
- Pitch angle
- Power production
- Power reference
- Rotor angular velocity
- Tower acceleration longitudinal

- Tower acceleration transverse
- Wind direction
- Wind speed.

Variables

Further, the SCADA server can hold real valued variables. These variables can be updated with 1 Hz based on measurements, using summation and multiplication. If $x \in \mathbf{R}^n$ are variables and $y \in \mathbf{R}^m$ are the measurements in the SCADA system, it is possible to update a variable as

$$x := Ax + By \tag{9.1}$$

where $A \in \mathbf{R}^{n \times n}$ and $B \in \mathbf{R}^{n \times m}$ are constant matrices [Ves11].

Update of Power Set-Points

In the feedback control, we want to dynamically update the power set-points P^i_{dem}. The SCADA system makes it possible to update the power set-points based on multiplication and summation of measurements [Ves11]. It is assumed that also variables can be used when updating power set-points. We can thus write the update of power set-points as

$$P^i_{\text{dem}} := c^T x + d^T y \tag{9.2}$$

where $c \in \mathbf{R}^n$ and $d \in \mathbf{R}^m$ are constants.

It is desired to design the controller, such that the power set-points are updated with a frequency of maximum 0.2 Hz. The reason is that a significant time delay is present when updating the power set-point. Further, a significant rate on the power set-point is expected. We assume that the time delay is 1 second, and that the rate limit is 100 kW/s, that is

$$\left| P^i_{\text{dem}} \frac{d}{dt} \right| \leq 100 \text{ kW/s}. \tag{9.3}$$

Distributed Control

It is noticed, that all the control is implemented on the SCADA server system. This means that the control must be implemented in a centralized manner, not distributed.

Model of the Vestas V90 Turbine

The controller design should be based on a model of the Vestas V90 3 MW wind turbine. As this model is not yet available, this work will be based on the National Renewable Energy Laboratory (NREL) wind turbine model, as previously used. Designing the controller based on this wind turbine will of course not give precise results, but it will be easy to change model when the Vestas V90 model is available.

In the next chapter we design a controller suitable for implementation on the server system at Thanet wind farm, taking the limitations present in this chapter into account.

Chapter 10

Control Strategy

In this section the overall control strategy for the experiments at Thanet wind farm is presented. The main focus of the control strategy is to make the controller implementable on the wind farm server at Thanet wind farm, which is accomplished by performing a number of simplifications. These simplifications are presented in this chapter.

10.1 Simplified Distributed Wind Farm Controller

The following sections will illustrate the simplifications made in order to accommodate limitations of the wind farm Supervisory Control And Data Acquisition (SCADA) system. Three main changes are made in the control strategy as follows.

- The distribution of the control is altered, so that most of the turbines in the wind farm operate individually.
- The feedback matrix is calculated offline and not updated during operation.
- Only the torque on the tower is included in the control objective.

The following sections describe the modifications in detail.

The main motivation for the simplifications is the difficulties introduced by the many limitations of the SCADA server system. The main limiting factor is the slow sampling time of 0.2 Hz. The problem of this slow sampling rate is easily illustrated by looking at the step response from a single wind turbine from the control input P^i_{dem} to the torque on the tower M_t. This step response is illustrated in Figure 10.1 (left). The figure reveals dynamics with a rise time of 1.4 s, indicating that the sampling frequency should be around 1 to 5 Hz [ÅW97, p. 66]. This large factor between the desired sampling frequency and the actual sampling frequency makes the desired type control difficult, which is one of the reasons for the simplifications.

Also figure 10.1 (right) illustrates the too slow control rate. The figure is similar to the one presented in part one in Figure 4.4 on page 21, and shows the tower bending in typical wind conditions. The red dots illustrate the rate of control, when controlling with 0.2 Hz, and illustrates that the sample rate is slower than the dynamics of the tower.

The scripts are found in

`matlab/nrel_freq_analysis/freq_analysis.m`

matlab/nrel/nrel_response.m

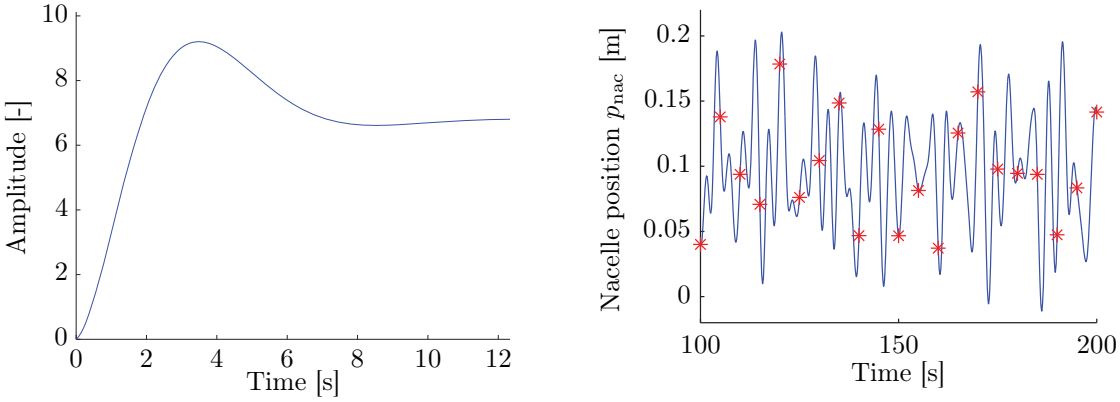

Figure 10.1: The left figure shows the torque tower M_t when applying a step in the control input P^i_{dem}. The right figure illustrates the sample rate of 0.2 Hz compared to the turbine dynamics.

10.2 Distribution of Controller

A major change is made in the distribution of the controller. Instead of coupling all the wind turbines through the global power constraint, it is chosen to let p out of the N turbines in the farm, operate individually. Each of these p wind turbines will individually seek to minimize their local fatigue, with no regard of the remaining turbines. Based on local state measurements, they will through state feedback control their power set-point P^i_{dem} to minimize the torque on the tower.

The remaining $N - p$ turbines will cooperate and make sure, that the global power constraint is honored. This concept is illustrated in Figure 10.2. The individually operating turbines $1 \dots p$ are shown to the right, each with a local feedback loop through feedback matrix L.

The sum of the set-points for the first p turbines $\sum_{i=1}^{p} P^i_{\text{dem}}$ is provided to the remaining $N - p$ wind turbines, which will make sure the power constraint is kept.

The reason for choosing this control strategy, is that this gives maximum freedom for the first p wind turbines regarding fatigue minimization. They only have to minimize their own fatigue load, regardless of the states of the remaining turbines. One such standalone wind turbine minimizing its own fatigue, is seen as the simplest case illustrating the benefits of wind farm control. If fatigue minimization of such a standalone wind turbine can not be obtained, we can not expect to obtain performance using a controller as described in the first part of this work.

Based on this, the sequel will focus on local control of one standalone wind turbine.

Note that the above described control strategy has a number of disadvantages. Both modularity and scalability is lost for the $N - p$ wind turbines. If a new turbine is added to the wind farm, the software on the $N - p$ wind turbines must be modified to include the new turbine.

Another disadvantage is that the $N - p$ wind turbines need not only communicate with the neighboring turbines, but with the whole wind farm, in order to ensure the global power constraint.

The above disadvantages destroy much of the whole background for the control strategy developed in part one. But we remind that the focus on this second part is to develop a controller,

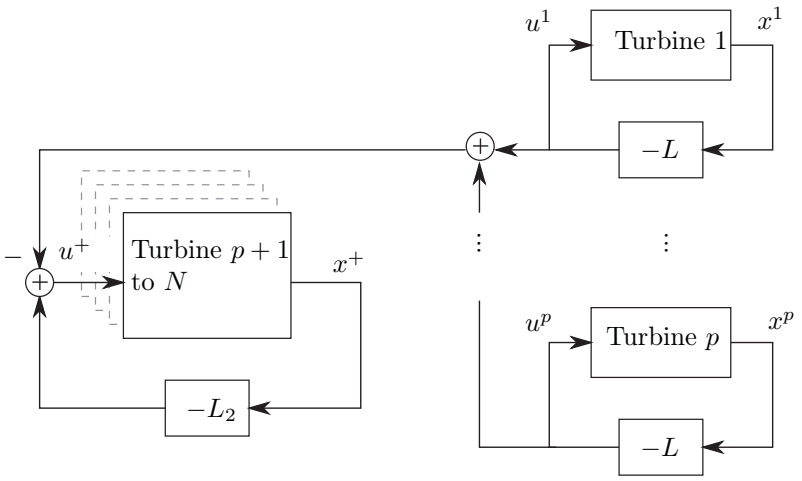

Figure 10.2: Illustration of the control strategy to be used in the experiments at Thanet. The first p turbines (to the right) operate individually, minimizing their own structural fatigue using their control variable $u^i = P^i_{\text{dem}} - \overline{P^i_{\text{dem}}}$. The remaining $N-p$ turbines (left) compensate for the individually operating turbines, and ensure that the global power constraint is honored.

that is implementable on Thanet wind farm, and still shows that farm control can enhance the performance of a wind farm.

10.3 Control Objective

To simplify the controller further, the objective is modified. In the control strategy in part one of this work, the objective was to minimize the rainflow count of the position of the nacelle p_{nac} and the rainflow count on the torsion of the shaft θ. This objective was achieved by minimizing the two norm of the torque on the tower and the torque on the shaft, respectively.

In the simplified controller, the objective is modified to only include the torque on the wind turbine tower, *i.e.* we let

$$y_t = C_c x_c + D_c u_t = M^i_t - \overline{M^i_t} \tag{10.1}$$

by letting the matrices C_c and D_c only contain the row corresponding to the tower torque, see Equation 5.8 on page 29.

There are a number of reasons for concentrating on only one output parameter. The reasons are:

- It will be easier to detect an increase in performance, when all effort is put into minimizing just one output. We remind that reduction of tower and shaft torque respectively are conflicting objectives.

- The tower displacement is directly measurable, making it easy to detect an increase in performance. The torsion of the shaft is not directly measurable, and would thus have to be reconstructed by filtering other measurements.

- Simulations show that at slower control frequencies, it is easier to control the tower bending, than the torsion on the shaft.

10.4 Linear Quadratic Control

As mentioned above, the aspect of modularity and scalability is lost, due to the new communication pattern. This means that there is no reason to update the feedback matrix online, as this was done to achieve modularity and scalability. Moreover it is noted, the calculations and memory necessary to update the feedback matrix exceed the limits of the SCADA system.

Therefore it is chosen to implement linear quadratic control in the wind farm, using a predetermined feedback matrix. A feedback controller must therefore be derived for the p individually operating wind turbines, feeding back the states to the power set-point. Similarly, a feedback matrix must be designed for the remaining $N - p$ cooperating turbines. This feedback matrix must assure that the power demand is honored.

In the sequel, we will only consider the controller for the p standalone wind turbines. In the next chapter we design this controller and evaluate the performance.

Chapter 11

Controller Design and Evaluation

This chapter presents the problem formulation of tower fatigue minimization of a standalone wind turbine. Following, a controller is designed, taking the limitations of the Supervisory Control And Data Acquisition (SCADA) server system into account. Finally the performance is examined through simulations.

11.1 Problem Formulation

Based on the setup described in last chapter, we develop a controller for one standalone wind turbine. This corresponds to the control of one of the p wind turbines, operating alone. We do not consider the remaining $N - p$ turbines.

In the following we present a problem formulation for fatigue minimization on a single wind turbine.

After removing the shaft torque as an output, we get the system

$$\begin{aligned} \dot{x}_c(t) &= A_c x_c(t) + B_c u_t(t) + B_{d,c} w_v(t) \\ y_t(t) &= C_c x_c(t), \end{aligned}$$

see 5.8.

The state of this system includes the wind turbine states and the wind state. The output is the small signal values of the torque on the tower while the input is the small signal value of the power set-point

$$x_c(t) = \begin{bmatrix} \beta(t) - \overline{\beta} \\ \omega_r(t) - \overline{\omega_r} \\ \omega_g^{\text{filt}}(t) - \overline{\omega_g^{\text{filt}}} \\ x_v(t) - \overline{x_t} \end{bmatrix} \qquad y_t(t) = M_t(t) - \overline{M_t} \qquad u_t(t) = P_{\text{dem}}(t) - \overline{P_{\text{dem}}}. \tag{11.1}$$

The term w is the uncontrollable noise input, due to the effect of the wind. Again we note that the white noise term has infinite energy, and therefore involves some idealization [GL00, p. 108].

The cost function is a trade-off between minimizing the control signal and minimizing the tower torque variation. We will formulate the cost in discrete time later in this chapter.

11.2 Accommodating System Limitations

The controller design must take the limitations previously described of the SCADA system into account. In the following we first treat the transport delay in the control signal, and the secondly we treat the rate limit of the power set-point.

Time Delay

A delay of 0.5-1 s is present when changing the power set-point of the wind turbines. By modeling this delay in the control variable, we can accommodate this delay in the controller design. As previously described, the wind turbine dynamics are given by

$$\dot{x}_c(t) = A_c x_c(t) + B_c u_t(t) + B_{d,c} w(t) \tag{11.2}$$

when the exogenous input is neglected and with no time delay is present. If a delay τ is introduced in the control variable, the dynamics become

$$\dot{x}_c(t) = A_c x_c(t) + B_c u_t(t-\tau). \tag{11.3}$$

This is difficult to handle in continuous time, as this is an infinite-dimensional system. When transforming the system to discrete time, however, it becomes a finite system. As the time delay $\tau \in [0.5,\ 1]$ s which is smaller than the sampling time constant $h = 5$ s, we know that the continuous system 11.3 can be transformed to a discrete version, by introducing a new state [ÅW97, p. 38]. With delay τ and sampling frequency h, the zero order hold sampling of the continuous system 11.3 becomes [ÅW97, p. 39]

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + B_d w(k) \\ y(k) &= Cx(k) \end{aligned}$$

where

$$A = \begin{bmatrix} A_1 & B_1 \\ 0 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} B_0 \\ I \end{bmatrix}, \qquad B_d = \begin{bmatrix} B_{d,c} \\ 0 \end{bmatrix}, \qquad C = \begin{bmatrix} C_c & 0 \end{bmatrix} \tag{11.4}$$

$$\begin{aligned} A_1 &= e^{A_c h} \\ B_0 &= \int_0^{h-\tau} e^{A_c} ds B_c \\ B_1 &= e^{A(h-\tau)} \int_0^{\tau} e^{A_c s} ds B_c. \end{aligned}$$

As is evident from the above system, the input $u(k)$ enters the system as the last element of the state vector, *i.e.*

$$x(k) = \begin{bmatrix} x_c(k) \\ u(k-1) \end{bmatrix}. \tag{11.5}$$

The time delay τ is hereby included in the discrete wind turbine model. Figure 11.1 shows the continuous system with and without delay, assuming a delay of 1 s. The figure shows the step response from the input $u = P^i_{\text{dem}} - \overline{P^i_{\text{dem}}}$ to the torque on the tower $y = M^i_t - \overline{M^i_t}$. In the same plot, the discretized systems are shown, both with and without the delay.

The script is found in

`matlab/thanet_state_feedback/standalone_turbine.m`

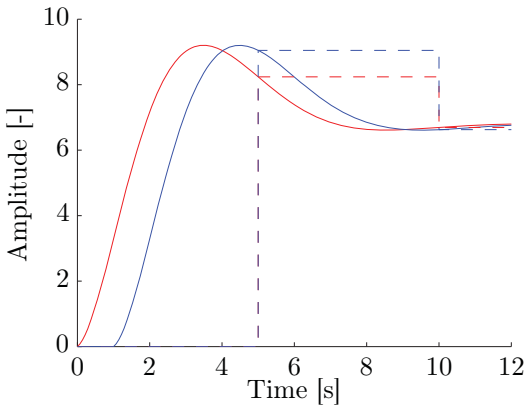

Figure 11.1: The continuous time model step responses from u to y with no delay (red) and with 1 s delay (blue) and corresponding discretized systems (dashed). The sampling frequency is 0.2 Hz.

Rate Limit and Saturation

The power set-point has given limits, depending on the wind velocity. For the Vestas V90 3 MW wind turbine, the lower limit is $P_l = 0.75$ MW, while the upper limit is $P_u = 3.00$ MW. A rate limit of 100 kW/s is further assumed.

As mentioned, the National Renewable Energy Laboratory (NREL) 5 MW wind turbine is used for controller design in this work, as the model of the Vestas V90 is not yet available. For this turbine, the limits are assumed to be $P_l = 0.75$ MW and $P_u = 5$ MW. We also assume a rate limit for this turbine of 100 kW/s.

In the design phase, we adjust the penalization of the control input $u = P^i_{\text{dem}} - \overline{P^i_{\text{dem}}}$, such that saturation and rate limit violations are avoided as far as possible.

11.3 Controller Design

We state the optimization problem for one standalone wind turbine. After introducing the new state, accommodating the time delay in the control variable, and after changing the cost function, we get the problem

$$\begin{array}{ll} \text{minimize} & J = \mathbf{E}(y^T Y y + u^T U u) \\ \text{subject to} & x(k+1) = Ax(k) + Bu(k) + B_d w(k) \\ & y(k) = Cx(k) + Du(k) \end{array} \tag{11.6}$$

where the variables are the states $x(k) \in \mathbf{R}^p$, the input $u(k) \in \mathbf{R}$ and the output $y(k) \in \mathbf{R}$. Further it is given that $\mathbf{E}(ww^t) = W$.

The system cost J penalizes the use of control signal through R, and the torque on the tower Q. Note that in this case with a single standalone wind turbine, Q and R are scalars, and could be replaced with a single trade-off parameter. It is however chosen, to keep the notation similar to the previous chapter.

The minimization of J is a trivial Linear Quadratic Regulator (LQR) problem, where the solution

is state feedback [ÅW97, p. 412]

$$u(k) = -Lx(k). \tag{11.7}$$

We find the feedback matrix L based on the system dynamics and the trade-off matrices using standard LQR methods.

As the wind state x_v is not measurable, and as it is assumed that the turbine measurements are corrupted by noise, a Kalman filter is designed. This filter estimates all states, based on the measurable states.

In order to design a Kalman filter, it is necessary to know how noise enters the system. The state noise is caused by the wind entering the system, which is already modeled. Additionally it is needed to know the measurement noise on all the measured signals, and it must be assumed that this noise is Gaussian. Let the measurable outputs be y_m, then

$$y_m(k) = C_m(k)x(k) + v(k). \tag{11.8}$$

The matrix C_m simply selects all states except the wind state, as these are measurable. The term $v(k)$ is the measurement noise.

Based on the covariance of the state noise w and the measurement noise v, the Kalman filter is designed based on standard methods. The states estimated by the Kalman filter are given by

$$\dot{x}_e(k) = Ax_e(k) + Bu_m(k) - K(y_m(k) - C_m x_e(k)) \tag{11.9}$$

where $y_m(k)$ is the measurement of the output, and $x_e(k)$ is the state estimation while K is the Kalman gain matrix [ÅW97, p. 431].

11.4 Performance Evaluation

Based on the above system, we evaluate the performance of the controller. We use the linearized continuous time model of the NREL wind turbine.

A wind field generated based on the wind spectra presented in Section 4.3 on page 22 is used as the uncontrollable input, and we close the loop using an LQR controller with appropriate weights Q, R. The controller is designed with a sample rate of $h = 5$ s, as previously described.

In the simulations the plant is the NREL wind turbine linearized for an operating point of $\bar{v} = 15$ m/s, and $\bar{P} = 3.0$ MW. A transport delay of 1 s and a rate limit of ± 100 kW/s are included in the model.

An example of a transient response of the control signal is illustrated in Figure 11.2. Here the effects of the slew-rate and the transport delay is clearly seen.

Assuming full state knowledge, the wind turbine controller developed for the operating point $\overline{v} = 15$ m/s, and $\overline{P} = 3.0$ MW is evaluated for a number of wind conditions. The reduction in fatigue is evaluated for each operating point. We denote the fatigue J_t and remind that only the fatigue on the tower is considered, *i.e.* $J_t = \mathbf{E}(yT^Y y)$.

The fatigue reductions in various operation points are illustrated in Table 11.1.

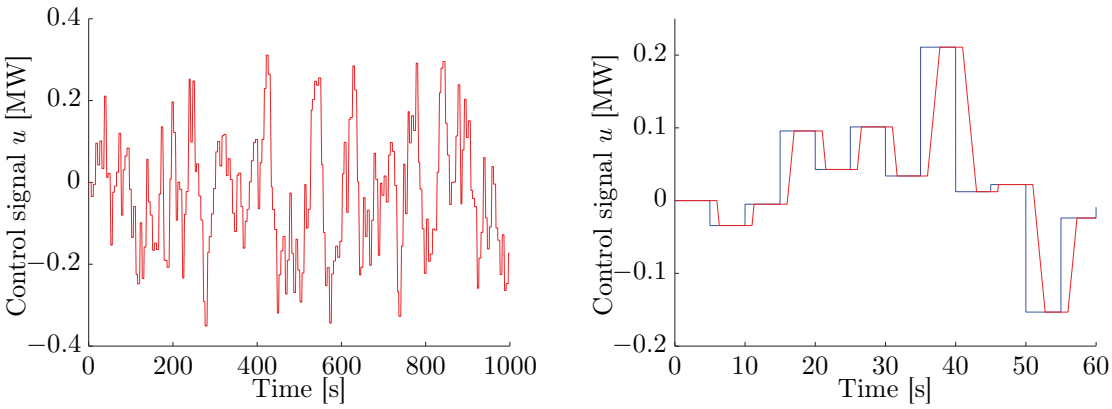

Figure 11.2: The left figure illustrates the use of control signal u, showing that the saturation limits are not violated. The right figure is a closeup of the first 60 s, showing the control signal u generated by the controller (blue), and the set-point that the wind turbine registers (red), after rate limit and a time delay.

Fatigue reduction [%]	$\overline{v} = 15$ m/s	$\overline{v} = 20$ m/s	$\overline{v} = 25$ m/s
$P_{\text{dem}} = 2.0$ MW	8	2	1
$P_{\text{dem}} = 3.0$ MW	11	4	2
$P_{\text{dem}} = 4.0$ MW	14	7	3

Table 11.1: The fatigue reduction of J_t when using LQR state feedback on a single wind turbine, assuming full state knowledge.

A similar table is generated, assuming that a Kalman filter has to be implemented to filter the states. Also the wind state is not measurable, and has to be estimated. Including this filter in the design, we get the decrease in fatigue as shown in Table 11.2.

Fatigue reduction [%]	$\overline{v} = 15$ m/s	$\overline{v} = 20$ m/s	$\overline{v} = 25$ m/s
$P_{\text{dem}} = 2.0$ MW	0	0	0
$P_{\text{dem}} = 3.0$ MW	2	0	0
$P_{\text{dem}} = 4.0$ MW	4	1	0

Table 11.2: The fatigue reduction of J_t when using LQR state feedback and Kalman state estimation on a single wind turbine.

The scripts for the values of the tables are found in

`matlab/thanet_state_feedback/standalone_turbine.m`

In the next chapter, we conclude on the results.

Part III

Epilogue

Chapter 12

Conclusion

In this work, we considered the control of a wind farm consisting of a large number of turbines. The basis of this wind farm control is the freedom that lies in the power distribution among the turbines, when there is more power available, than demanded. The focus of this work was to distribute the power demand to the turbines in the farm dynamically, such that the total fatigue on the turbines in the farm is minimized. At the same time the power demand to the farm must be met. In the first part of this work we showed that this power distribution can be achieved by a distributed controller. Moreover, we demonstrated that this controller can be designed such that it is both modular and scalable. The second part concerned the upcoming experiments at Thanet wind farm. In this part of the work, we developed a new controller, taking the limitations of the wind farm server system at Thanet into account.

12.1 Modular Distributed Wind Farm Control

In the first part of the thesis, we designed a distributed modular and scalable wind farm controller. The basis of the controller was the coupling of the wind turbines through the total power demand to the wind farm. The coupling of the turbines through the wind field was neglected.

We showed that a freedom lies in the distribution of the total wind farm power demand to the individual turbines in the wind farm. When there is an excess of wind energy available, we are free to distribute the power demand among the wind turbines, as long as the farm power demand is met.

Based on this freedom, a controller was designed that dynamically generates set-points for the turbines in the farm, using turbine measurements. The objective of the controller was to reduce the total fatigue on the turbines in the farm, while still honoring the farm power demand. Fatigue on both turbine tower and generator shaft were taken into account.

Further we showed that the designed controller is both modular and scalable. This gives the desired properties, that the controller on each wind turbine in the farm is identical. Also it is possible to add or remove turbines from the wind farm, without changing the controller on the remaining turbines.

The basis of the controller is distributed state feedback. Each turbine controls its power set-point, by feeding back own and neighboring measurements. Moreover, each turbine updates the local control law iteratively, based on local and neighboring measurements.

By simulations in a linear wind farm model, this iterative controller was verified. The simulations showed that the modular and scalable design indeed is implementable, and that the distributed

controller converges. The controller was also examined in various wind fields to demonstrate robustness against changes in the wind field.

Further simulations were performed in a realistic wind farm environment consisting of 10 NREL wind turbine models. The simulations confirmed the functionality of the distributed control strategy. Also these simulations revealed, that fatigue reductions in the magnitude of 50 % for the shaft and 10 % for the tower could be achieved.

Based on this we conclude that the area of wind farm control is a promising approach to fatigue reduction in wind farms. With the only requirements that the turbines are allowed to communicate with neighbors and change their power set-points dynamically, we can reduce the fatigue on both the turbine tower shaft significantly. Moreover we conclude, that this controller can be designed as both modular and scalable such that turbines can be removed or added to the farm, without altering the software on the remaining turbines.

12.2 Distributed Controller for Thanet Wind Farm

In the second part of the thesis, we considered upcoming wind farm experiments, planned to be performed at Thanet wind farm in the summer of 2011. The goal of the experiments is to demonstrate the concept of fatigue reduction through dynamic power distribution.

By examination of the wind farm server at Thanet wind farm, it was found that both Kalman filtering and state feedback can be implemented on the turbines.

The examination of the turbines at Thanet wind farm however also revealed a number of limitations. Updating the power set-points is characterized by both a rate limit and a significant time delay. Also, it is only possible to update the power set-points with a relatively slow frequency. Finally the server system only offers a very limited number of variables and operations.

Based on the strict limitations, a simplified farm control strategy was designed. The basic idea of this strategy is to allow most of the turbines in the farm to operate individually. Each of these individual turbines minimize their fatigue by power set-point control, feeding back local measurements. The remaining turbines ensure that the farm power demand is met.

We showed that the modified controller is implementable on Thanet wind farm. We further showed, that the modifications give the benefit that the concept of wind farm control is reduced to observing one single standalone wind turbine.

A number of simulations were performed to evaluate if performance could be obtained when the limitations of the system were taken into account. These simulations were conducted on a linearized model of the National Renewable Energy Laboratory (NREL) wind turbine model, as a model of the turbines at Thanet was not available. A transport delay and rate limitation were added to the model, and the controller operated with the sampling rate allowed by the system. Assuming perfect measurements, fatigue reductions in the magnitude of 5 % were observed. When including a Kalman filter to estimate the states, this performance was reduced to 0-1 %.

Assuming that the used model corresponds well to the turbines at Thanet wind farm, we conclude that it is not possible to achieve fatigue reduction using dynamic power distribution at Thanet wind farm. The limitations introduced by the current wind farm server system make the controller too slow compared to the wind farm dynamics.

Appendix A

Rainflow Counting Algorithm

The first step in the rainflow algorithm, is to transform the signal from the measured signal, to a signal consisting solely of the turning points, as illustrated in Figure A.1. The plot now consists of peaks and valleys, labeled A through I. Let a *range* be the difference between a peak and a valley, as illustrated in Figure A.1, where the range between valley G and peak H is illustrated and denoted $|G - H|$. This is the basis of the algorithm. The algorithm from [AST05] is:

Let X denote the range under consideration, and Y the previous range. Let S be the starting point. With the labels as in Figure A.1, the algorithm would thus start with $S = A$, $Y = |A - B|$ and $X = |B - C|$. The algorithm is:

1. Read next peak or valley. If out of data, go to 6.
2. If less than 3 points are read, go to Step 1. Form range X and Y of the three latest peaks/valleys not discarded.
3. Compare the ranges X and Y, if $X < Y$ go to Step 1, else if $X \geq Y$ go to Step 4.
4. If the range Y consists the starting point S, go to Step 5. Else count the range Y as one cycle, and discard the peak/valley set in Y and go to Step 2.
5. Count the range Y as one half cycle, discard the first peak or valley in Y, move the starting point S to the next peak/valley, and go to Step 2.
6. Count each range that has not been counted yet as one half cycle.

Rainflow Counting Example

The example presented in [AST05] is presented here, to clarify the rainflow algorithm.

First the sampled signal, Figure A.1 (right) is transformed to a signal of only the turning points, Figure A.1 (left). Starting from the start of the history, with turning point A, the signal is processed as follows. It is reminded that S is the starting point, X is the range under consideration while Y is the previous range.

1. $S := A$, $Y := |A - B|$, $X := |B - C|$
   $X > Y$, $S \in Y$
   $Y = |A - B|$ is counted as a half cycle, and A is discarded.

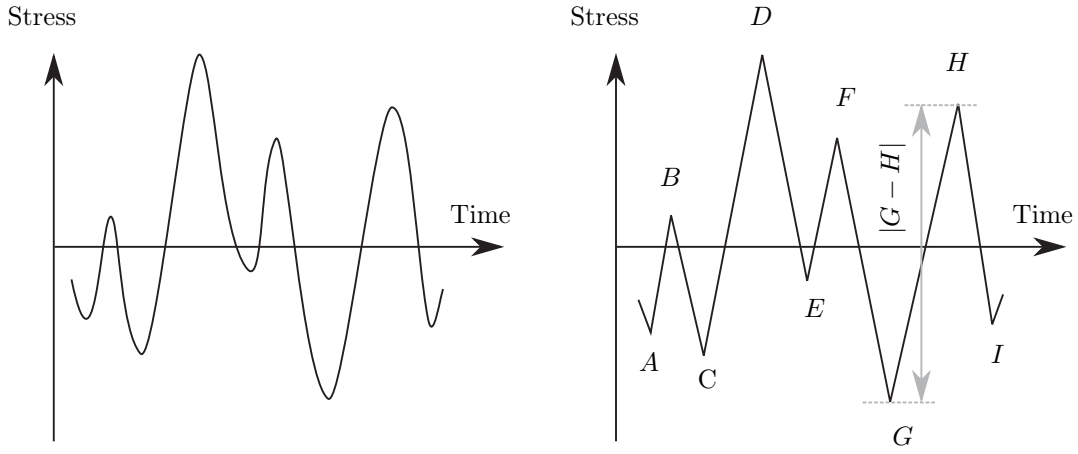

Figure A.1: Stress measurement (left) converted to turning points (right). Figure from [AST05].

2. $S := B$, $Y := |B - C|$, $X := |C - D|$
   $X > Y$, $S \in Y$
   $Y = |B - C|$ is counted as a half cycle, and B is discarded.

3. $S := C$, $Y := |C - D|$, $X := |D - E|$
   $X < Y$, so next peak/valley is read (peak E).

4. $S := C$, $Y := |D - E|$, $X := |E - F|$
   $X < Y$, so next peak/valley is read (valley G).

5. $S := C$, $Y := |E - F|$, $X := |F - G|$
   $X > Y$, $S \notin Y$
   $Y = |D - E|$ is counted as one cycle, and E and F are discarded.

6. $S := C$, $Y := |C - D|$, $X := |D - G|$
   $X > Y$, $S \in Y$
   $Y = |C - D|$ is counted as a half cycle and C is discarded

7. $S := D$, $Y := |D - G|$, $X = |G - H|$
   $X < Y$, so next peak/valley is read (valley I).

8. $S := D$, $Y := |G - H|$, $X = |H - I|$
   $X < Y$, end of data.

9. $|D - G|$, $|G - H|$, $|H - I|$ are unaccounted for, and therefore all counted as a half cycle

Appendix B

List of Acronyms

LQR Linear Quadratic Regulator

NREL National Renewable Energy Laboratory

SCADA Supervisory Control And Data Acquisition

Bibliography

[Aeo08] Aeolus. Aeolus distributed control of large scale offshore wind farms. Fact Sheet, September 2008. http://ict-aeolus.eu/pub/fp7-fact-sheet-aeolus_en.pdf.

[Aeo10a] Aeolus. Simplified nrel5mw turbine. Webpage, August 2010. Detailed description of NREL 5 MW Wind Turbine. http://www.ict-aeolus.eu/SimWindFarm/model-turbine.html.

[Aeo10b] Aeolus. Simwindfarm. Webpage, February 2010. Fast wind farm simulation environment for development of wind farm control algorithms. http://www.ict-aeolus.eu/SimWindFarm/.

[Aeo11] Aeolus. Experiment description. Aeolus Deliverable, April 2011.

[AST05] ASTM. Astm e 1049-85, standard practices for cycle counting in fatigue analysis. *Designation: E 1049 85*, Reapproved 2005.

[ÅW97] K. Åström and B. Wittenmark. *Conputer-Controlled Systems*. Prentice Hall, 3rd edition, 1997. ISBN 0-13-314899-8.

[BJS+11] B. Biegel, M. Juelsgaard, J. Stoustrup, M. Kranning, and S. Boyd. Wind turbine pitch optimization. *Submitted to The 2011 IEEE Multi-Conference on Systems and Control*, 2011.

[Boy10] S. Boyd. Primal and dual decomposition. Lecture Notes, 2010. Stanford University, http://www.stanford.edu/class/ee364b/lectures/decomposition_slides.pdf.

[BSJB01] T. Burton, D. Sharpe, N. Jenkins, and E. Bossanyi. *Wind Energy Handbook*. Wiley, 2001.

[Bul01] S. R. Bull. Renewable energy today and tomorrow. In *Proceedings of the IEEE, vol. 89, no. 8*, August 2001.

[BV04] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 1st edition, 2004. ISBN 978-0-521-83378-3.

[Cou08] Ian Couchman. Plan for performance assessment. *Aeolus deliverable 5.5*, April 2008.

[GL00] T. Glad and L. Ljung. *Control Theory, Multivariable and Nonlinear Methods*. Taylor & Francis, 1st edition, 2000. ISBN 0-7484-0878-9.

[GSK+10] Jacob Deleuran Grunnet, Mohsen Soltani, Torben Knudsen, Martin Kragelund, and Thomas Bak. Aeolus toolbox for dynamic wind farm model, simulation and control. In *Proc. of the 2010 European Wind Energy Conference*, 2010.

[Ham06] K. Hammerum. A fatigue approach to wind turbine control. 2006. Master's Thesis prepared at Vestas Wind Systems.

[HSBF05] A. D. Hansen, P. Sørensen, F. Blaabjerg, and I. Florin. Grid support of a wind farm with active stall wind turbines and ac grid connection. *Wind Engineering*, (9):341–359, 2005.

[JBMS09] J. Jonkman, S. Butterfield, W. Musial, and G. Scott. Definition of a 5-mw reference wind turbine for offshore system development. Technical report, National Renewable Energy Laboratory, 2009.
`http://www.nrel.gov/wind/pdfs/38060.pdf`.

[JPP08] M. Jelavić, V. Petrović, and N. Perić. Individual pitch control of wind turbine based on loads estimation. *Industrial Electronics, 2008. IECON. 34th Annual Conference of IEEE*, November 2008.

[KBS09] Torben Knudsen, Thomas Bak, and Mohsen Soltani. Distributed control of large-scale offshore wind farms. In *European Wind Energy Conference and Exhibition (EWEC) 2009*, Marseille, France, 2009.

[Lue84] D. G. Luenberger. *Linear and Nonlinear Programming*. Addison-Wesley, 2nd edition, 1984. ISBN 0-201-15794-2.

[MMR11] Daria Madjidian, Karl Mårtensson, and Anders Rantzer. A distributed power coordination scheme for fatigue load reduction in wind farms. *Submitted to American Control Conference, San Francisco*, June 2011.

[MR09] Karl Mårtensson and Anders Rantzer. Gradient methods for iterative distributed control synthesis. In *Proceedings of the 48th IEEE Conference on Decision and Control*, Shanghai, China, December 2009.

[MR10] Karl Mårtensson and Anders Rantzer. Sub-optimality bound on a gradient method for iterative distributed control synthesis. In *Proc. 19th International Symposium on Mathematical Theory of Networks and Systems*, Budapest, Hungary, July 2010.

[NCVT10] S. Nourdine, H. Camblong, I. Vechiu, and G. Tapia. Comparison of wind turbine lqg controllers using individual pitch control to alleviate fatigue loads. *18th Mediterranean Conference on Control & Automation*, June 2010.

[Ofg09] Ofgem. Thanet offshore transmission assets. Technical report, 2009.
`http://www.ofgem.gov.uk/Networks/offtrans/rott/Documents1/Thanet%20-%20Project`

[Ran09] Anders Rantzer. Dynamic dual decomposition for distributed control. In *Proceedings of American Control Conference*, St. Louis, June 2009.

[SF01] R. I. Stephens and H. O. Fuchs. *Metal Fatigue in Engineering*. John Wiley & Sons, Inc., 2nd edition, 2001. ISBN 0-417-51059-9.

[SJBMV] V. Spudić, M. Jelavić, M. Baotić, and N. Perić M. Vašak. Deliverable 1.6: Plan for validation of control strategies. Public Aeolus Deliverables.

[SJBMV07] V. Spudić, M. Jelavić, M. Baotić, and N. Perić M. Vašak. Deliverable 3.3: Reconfigurable control extension. Public Aeolus Deliverables, 2007. University of Zagreb.

[SKB09] Mohsen Soltani, Torben Knudsen, and Thomas Bak. Modeling and simulation of offshore wind farms for farm level control. In *European Offshore Wind Conference and Exhibition (EOW) 2009*, Stockholm, Sweden, 2009.

[Spu] V. Spudić. Hierarchical wind farm control for power/load optimization. *Aeolus Deliverable*.

[Sti08] M. Stiebler. *Wind Energy Systems for Electric Power Generation*. Springer, 2008.

[Swi10] John Swigart. Optimal controller synthesis for decentralized systems. Ph.D. Dissertation, Stanford University, December 2010. http://junction.stanford.edu/~jswigart/publications/swigart_phd_dissertation.pdf.

[Ves11] Vestas. Aeolus test campaign at thanet offshore wind farm, uk. Technical report, Vestas Wind Systems, 2011.

[WWB10] H. Wang, W. Wang, and L. Bin. Application of individual pitch controller for flicker reduction on variable speed wind turbines. *Power and Energy Engineering Conference (APPEEC), 2010 Asia-Pacific*, March 2010.

[XXZ+08] Y. Xingjia, W. Xiaodong, X. Zuoxia, L Yingming, and L. Jun. Individual pitch control for variable speed turbine blade load mitigation. *Sustainable Energy Technologies, ICSET 2008. IEEE International Conference.*, November 2008.