A Fatigue Approach to Wind Turbine Control

Keld Hammerum

Kongens Lyngby 2006

Technical University of Denmark
Informatics and Mathematical Modelling
Building 321, DK-2800 Kongens Lyngby, Denmark
Phone +45 45253351, Fax +45 45882673
reception@imm.dtu.dk
www.imm.dtu.dk

Summary

Due to the turbulent nature of wind, the structural components of a wind turbine are exposed to highly varying loads. Therefore, fatigue damage is a major consideration when designing wind turbines.

The control scheme applied to the wind turbine affects a subset of the loads. In this report, the basics of wind turbine control are outlined, and a summary of traditional fatigue damage estimation is given, including a treatment of SN curves and rainflow counting procedures.

Spectrally based techniques providing approximate results for the fatigue damage estimate are presented. These methods describe the fatigue damage as a function of the spectral moments of the load history. This motivates the development of an efficient algorithm for computation of spectral moments of linear, stochastic processes.

These methods are combined to provide efficient evaluation of a turbine performance measure taking into account fatigue damage. An application of this cost function is demonstrated through numerical optimisation of a wind turbine controller, based directly on fatigue damage design objectives.

Resumé

På grund af vindens turbulente natur udsættes en vindmølles strukturelle komponenter for stærkt varierende belastninger. Derfor er udmattelseslaster en væsentlig designparameter for moderne vindmøller.

Vindmøllens regulator har indflydelse på en delmængde af disse belastninger. I denne rapport gennemgås grundlæggende vindmølle-regulering, og traditionelle metoder til analyse af udmattelseslaster behandles—heriblandt SN-kurver og rainflow counting metoder.

Spektralt baserede metoder til approksimativ bestemmelse af estimatet for udmattelsesrate præsenteres. Disse metoder benytter sig af de spektrale momenter for den påvirkende belastning, hvilket motiverer udviklingen af en effektiv algoritme til beregning af spektrale momenter for lineære, stokastiske processer.

Disse metoder kombineres til at opnå effektiv beregning af en tabsfunktion, hvor den forventede udmattelseslast indgår. Anvendelse heraf demonstreres gennem numerisk optimering af en vindmølleregulator, hvor minimering af udmattelseslasten indgår direkte som designmål.

Preface

This Master's thesis was prepared at Vestas Wind Systems, Randers, Denmark as a requirement for acquiring the Master's degree in engineering at the instute for Informatics and Mathematical Modelling, the Technical University of Denmark.

The thesis deals with wind turbine controller design taking into account fatigue damage.

Randers, December 2006

Keld Hammerum

Acknowledgements

I would like to thank all the people in the Loads, Aerodynamics and Control group at Vestas Wind Systems for supporting this project. Especially, I thank my supervisor Ph.D, M.Sc.EE. Per Brath for encouraging support and guidance regarding project progress and goals.

Thanks goes to M.Sc.Eng. Erik Carl Miranda and M.Sc.Eng. Rasmus Svendsen for their daily contributions to the understanding of wind turbine dynamics and for their patience when elaborating on topics from mechanical engineering.

Also a special thank to M.Sc.Eng. Bo J. Pedersen for numerous fruitful discussions on fatigue loads as well as general aspects of the analysis of dynamic systems.

Finally, a very special thank to my supervisor, Assoc. Prof. Niels Kjølstad Poulsen, IMM, DTU, for providing qualified and dedicated guidance on any matter during the project period.

Contents

Chapter 1

Introduction

This chapter gives a brief introduction to wind energy and wind turbine design.

1.1 Wind energy

In Denmark, 20% of the electrical power comes from wind energy. Worldwide, the number is approximately 0.6%. Further increase of this proportion requires wind power production to be as economical attractive as conventional power production. Some claim that, if environmental costs are included, wind energy *is* cheaper than conventional power production. It is hard, though, to convince power companies that they should invest in the apparently more expensive wind power plant because of environmental benefits. Therefore, a further decrease in wind energy costs is crucial to further growth.

1.2 Wind turbine design

Wind turbine design is ultimately governed by economic considerations. That is, turbines must be constructed in such a way as to optimise the profit for the

manufacturer. As stable, large profits can only be obtained through long-term customer satisfaction, reliability is a major concern.

These considerations constitue the basic trade-off in wind turbine design: production costs must be low to increase the contribution margin for the manufacturer. This naturally advocates for low-weight, low-cost turbine components. On the other hand, the demand for reliability advocates for heavy, robust structures designed using high partial factors.

To find the optimal trade-off, the overall turbine performance should ideally be expressed by a single number; a cost J that would be the sum of a large number of performance measures, i.e:

$$J = \sum_i w_i J_i,$$

where w_i denotes the weighting of the performance measure J_i. The determination of the weights w_i would depend upon not only technical matters, but also on economic considerations regarding wind turbine maintenance costs, risk analysis, etc.

The set of cost contributions J_i would include things like power efficiency, power quality, and the large class of mechanical load measures, including extreme loads and fatigue loads as well as thermal loads.

A subset of these performance measures are affected by the wind turbine control scheme. As an example, drive train vibrations will depend highly on the controller, whereas the loads inflicted by waves on an off-shore turbine are unlikely to be significantly affected by the control scheme.

Wind turbine controller design calls for a number of compromises between conflicting objectives—for example the well-known pitch activity vs. speed control trade-off. As controller complexity grows with the advent of e.g. split-pitch controllers, the available trade-offs might become less obvious.

This calls for a framework providing a means for numerical optimisation of the controller. Such a framework would consist of a cost function formulation along with algorithms capable of solving the optimisation problem of minimising the cost, with the controller parameters (or structure) being the decision variables:

$$U^* = \underset{U}{\operatorname{argmin}}\, J(U),$$

where U^* denotes the optimal controller.

A complete parameterization of the wind turbine control scheme leads to a high-dimensional optimisation problem, which, in turn, implies a very large number of

cost funtion evaluations. Therefore, efficient cost function evaluation is crucial.

The cost function evaluation would ideally consist of a complete analysis of all operation scenarios using aeroelastic simulation tools such as FLEX or HAWC. Basing a numerical optimisaton scheme on such a cost function evaluation is considered unrealistic with present-day computational resources. Therefore, techniques for faster cost function evaluation (or estimation) are needed.

This thesis concentrates on efficient evaluation (or estimation) of the cost function contributions that arise from fatigue damage. The ultimate goal is to provide management-level tuning knobs that allow system designers to tune the controller directly in terms of expected component lifetimes.

For further readings on general wind turbine design, the reader is referred to [BSJB01]. For a thorough treatment of probabilistic wind turbine design and fatigue, [Vel06] is recommended.

1.3 Thesis structure

In chapter 2, a model describing the dynamics of the wind turbine is presented, including a linearisation scheme and a discussion of neglected dynamics.

Chapter 3 outlines the basics of wind turbine control. Basic issues in wind turbine control are demonstrated through the design of a hybrid control scheme. Further, a flexible LQI design is used for demonstrating some of the trade-offs inherent to wind turbine controller tuning.

An introduction to fatigue damage estimation is given in chapter 4. This includes a treatment of SN curves and Palmgren-Miner's damage rule. In addition, the concept of rainflow counting is introduced along with the so-called four-point algorithm used for extracting the equivalent cycles from a stress history.

The lack of closed-form expressions for the fatigue damage has motivated works on approximations that expresses the fatigue damage as a function of the spectral moments of the damage-inflicting stress history. In chapter 5, the works of Rychlik and Benasciutti are summarised, leading to a compact fatigue damage estimate based on four spectral moments.

In chapter 6, the properties of spectral moments in linear systems are investigated, resulting in an algorithm that efficiently computes the spectral moments of linear, stochastic processes.

Chapter 7 demonstrates how the algorithms are combined to allow for efficient cost function evaluation in a numerical optimisation of a wind turbine controller taking into account fatigue loads.

Finally, the conclusions of the work along with suggestions for further works are found in chapter 8.

1.4 Notation

A subset of the notation used in the thesis is summarised in table 1.1.

Symbol	Usage
$\mathbb{R}$	The set of real numbers.
$\mathbb{C}$	The set of complex numbers.
$\Re(x)$	Real part of x.
$\Im(x)$	Imaginary part of x.
λ_m^x	The mth spectral moment of the process $x(t)$.
k	Wöhler coefficient.
K	Material constant defining SN curve.
$c_{v,x}$	Variability coefficient for the variable x.
s_θ	Stress level in the drive train.
s_z	Stress level in the tower.
λ	Tip speed-ratio. Eigenvalue.
d_θ	Damage rate for drive train.
d_z	Damage rate for tower.
d_β	Damage rate for the pitch bearings.
C_θ	Proportionality constant: $s_\theta = C_\theta \theta$.
C_z	Proportionality constant: $s_z = C_z z$.
C_β	Proportionality constant: $d_\beta = C_\beta \sqrt{\lambda_2^\beta}$.
Λ_x	Benasciutti damage rate approximation applied to x.

Table 1.1: Notation.

Chapter 2

Wind turbine modelling

To analyse the dynamics of the wind turbine and provide a basis for wind turbine controller design, a mathematical model of the turbine is needed. This chapter describes such modelling, including a linearisation scheme for the inherently non-linear turbine model. Finally, it gives a discussion of neglected dynamics.

2.1 Wind turbine components

The majority of commercially operated wind turbines are three-axis horisontal-axis wind turbines. As depicted in figure 2.1 such turbine consists of a nacelle mounted on a tower. The nacelle contains the key components of the wind turbine, including the gearbox, the electrical generator, and the main shaft providing the mechanical interface to the hub carrying the blades.

The blades are attached to the hub using bearings in order to allow the blades to be rotated around their own axis. The blade angle is referred to as the *pitch angle*.

Further, a yawing system allows the nacelle and the rotor unit to be turned against the wind. A schematic of a commercial wind turbine is shown in figure 2.2.

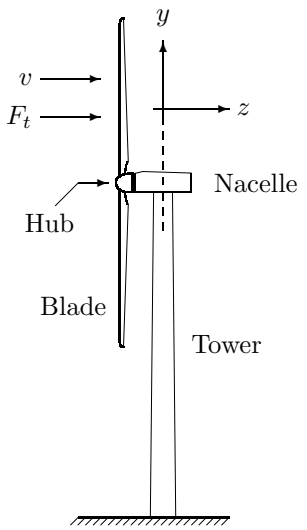

Figure 2.1: A horisontal-axis wind turbine (HAWT).

The interactions between the different components are depicted in figure 2.3. When the wind interacts with the blade aerodynamics, it will excert a torque T_a on the rotor, which is transferred to the generator via the *drive train*. The drive train consists of a low-speed shaft connecting the hub to the gearbox, and a high-speed shaft connecting the gearbox to the generator.

Further, the wind will excert a thrust force F_t on the tower, causing a deflection z of the tower structure.

Note that the wind speed v_r seen by the rotor plane is the incoming wind speed v superimposed by the nacelle velocity $\dot{z}$ resulting from the tower being deflected by the wind:

$$v_r = v - \dot{z}.$$

2.2 Wind modelling

Wind is chaotic in nature. A complete spectrum of the wind speed spans several decades, including seasonal variations on a yearly scale, diurnal variations, and the fastest variations—denoted turbulence—described in minutes and seconds. As the slowest dynamics of a wind turbine should be measured in seconds, the

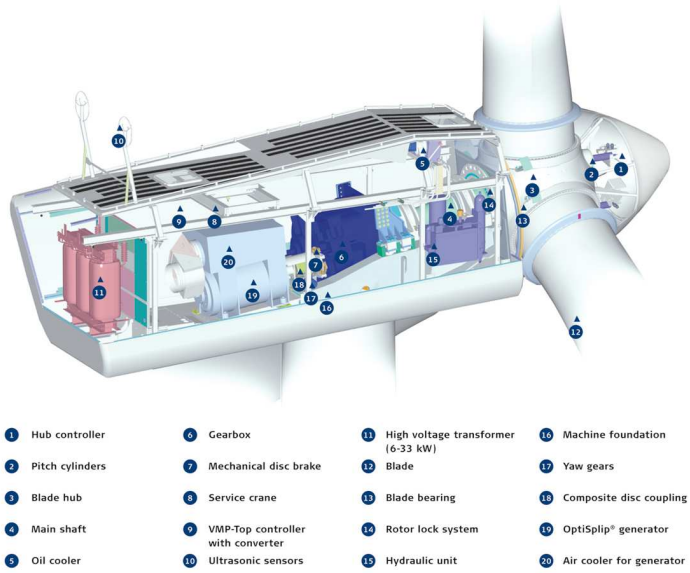

Figure 2.2: A schematic view of the inside of a Vestas V80-1.8MW nacelle. The high-speed shaft referred to in section 2.4 is the shaft betweeen the gear-box and the generator.

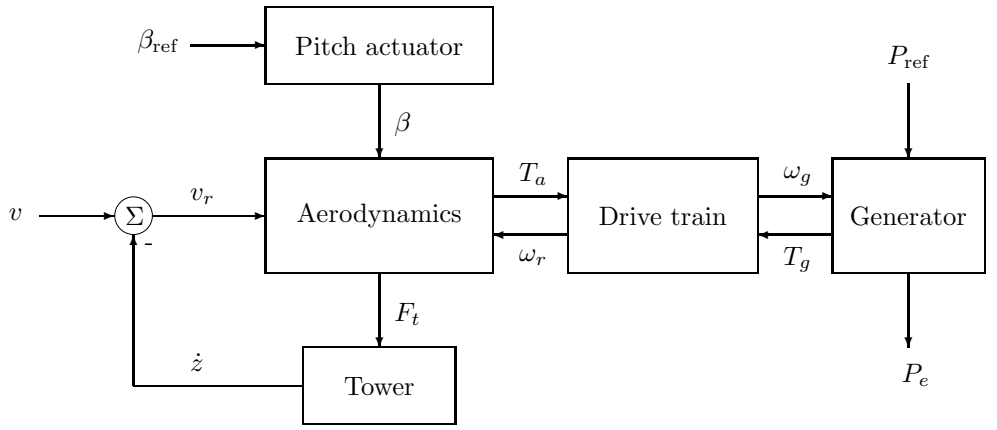

Figure 2.3: Interactions in the wind turbine model. Blade pitch angle reference β_{ref} and power reference P_{ref} are controllable inputs. Notice how nacelle velocity $\dot{z}$ affects the wind speed v_r seen by the rotor.

diurnal as well as the yearly variations can be considered as slow changes in the mean value for the wind experienced by the wind turbine. Thus, we will describe the wind speed v as a mean wind speed $\bar{v}$ pertubed by a turbulent contribution, $\tilde{v}$:

$$v = \bar{v} + \tilde{v}.$$

This notion leaves the task of modelling the stochastic, turbulent part. The following approach was suggested in [Knu83, Ma97].

Detailed studies of turbulence spectra result in irrational spectral descriptions. As a linear, stochastic model of the turbulence is desired, the irrational spectrum of the turbulence is approximated by a rational spectrum. A possible approximation of the turbulence spectrum $S_{\tilde{v}}(\omega)$ is given by

$$S_{\tilde{v}}(\omega) = \frac{k^2}{(1+\tau_1^2\omega^2)(1+\tau_2^2\omega^2)}, \tag{2.1}$$

which corresponds to the turbulence being modelled as a unity intensity white noise process filtered by the stable filter

$$H(s) = \frac{k}{(1+\tau_1 s)(1+\tau_2 s)}. \tag{2.2}$$

We will describe the turbulence process with the equivalent state-space description

$$\begin{pmatrix} \dot{\tilde{v}} \\ \ddot{\tilde{v}} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -\frac{1}{\tau_1\tau_2} & -\frac{\tau_1+\tau_2}{\tau_1\tau_2} \end{pmatrix} \begin{pmatrix} \tilde{v} \\ \dot{\tilde{v}} \end{pmatrix} + \begin{pmatrix} 0 \\ \varepsilon \end{pmatrix},$$

where ε is a white noise process with intensity $\left(\frac{k}{\tau_1\tau_2}\right)^2$.

The parameters k, τ_1, and τ_2 result from fitting (2.1) to the irrational spectrum of the detailed turbulence model, and are functions of the mean wind speed as depicted in figure 2.4. The turbulence variance $\sigma_{\tilde{v}}^2$ is found by integrating (2.1):

$$\sigma_{\tilde{v}}^2 = \frac{1}{2\pi}\int_{-\infty}^{\infty} S_{\tilde{v}}(\omega)\, d\omega = \frac{1}{2}\frac{k^2}{\tau_1+\tau_2}$$

and increases with the mean wind speed as shown in figure 2.5.

2.3 Aerodynamics

When wind passes the wind turbine rotor plane, part of the kinetic energy in the wind is transferred to the rotor. The power P_a obtained by the rotor is

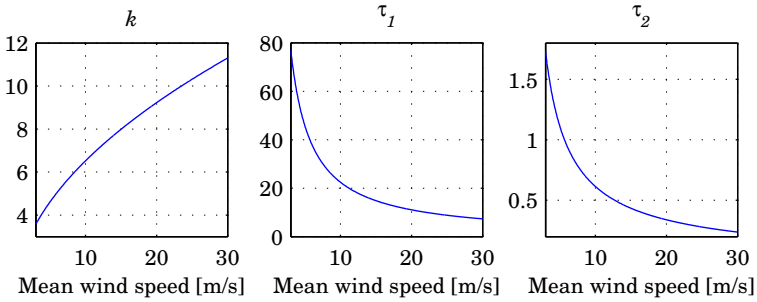

Figure 2.4: Parameters in the turbulence model as functions of the mean wind speed. Notice the increasing gain k and the decreasing time constants τ_1 and τ_2, cf. figure 2.5.

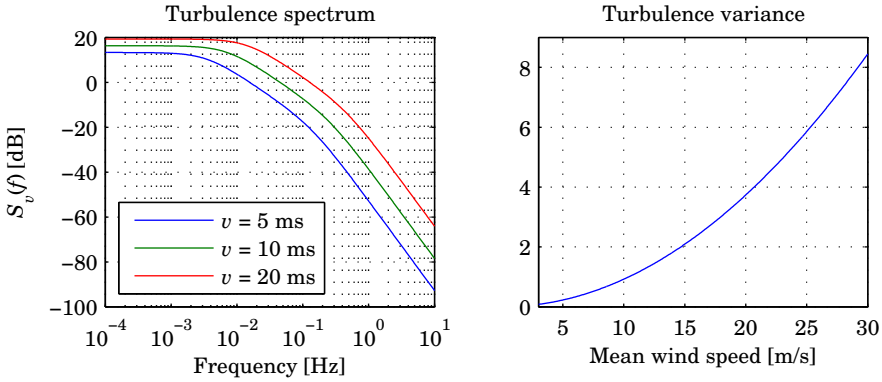

Figure 2.5: Left: The turbulence bandwidth increases with the mean wind speed, cf. the decreasing time constants shown in figure 2.4. Right: Turbulence variance as function of the mean wind speed.

given by

$$P_a = \frac{1}{2}\rho\pi R^2 v_r^3 C_P(\lambda, \beta), \tag{2.3}$$

where ρ is the air density, R is the rotor radius, and C_P is the power efficiency coefficient. The quantity C_P has a theoretical upper limit—known as the Betz limit—of $16/27 = 0.59$. That is, at most 59% of the energy in the wind can be extracted by the turbine. Further, notice the cubic relationship between wind speed v_r and the power.

In addition to delivering power to the wind turbine, the wind will excert a thrust on the rotor plane—that is, a force on the rotor in the fore-aft direction. The thrust force F_T is given by

$$F_T = \frac{1}{2}\rho\pi R^2 v_r^2 C_T(\lambda, \beta). \tag{2.4}$$

As indicated, the coefficients C_P and C_T are both functions of the tip speed-ratio λ and the blade pitch angle β, with tip speed-ratio defined as[1]

$$\lambda \equiv \frac{\omega_r R}{v_r}. \tag{2.5}$$

In figures 2.6 and 2.7 the C_P and C_T coefficients are plotted as functions of the tip speed-ratio λ and the pitch angle β. We will denote the global maximum on the C_P curve as $C_P^* = C_P(\lambda^*, \beta^*)$.

Finally, as the rotor power P_a and rotor torque T_a are related to rotor angular rotational speed ω_r as $P_a = T_a\omega_r$, we have for the aerodynamic torque:

$$T_a = \frac{1}{\omega_r}\frac{1}{2}\rho\pi R^2 v_r^3 C_P(\lambda, \beta). \tag{2.6}$$

2.4 Drive train

The drive train will be modelled as proposed in [CO04], where the structural model depicted in figure 2.8 is suggested. The model consists of the following components:

- The combined rotational moment of inertia I_r of the rotor and low-speed shaft.

[1] Note that the inverse definition of tip speed-ratio is also found in the literature. Here, the definition used in [BSJB01] is adopted.

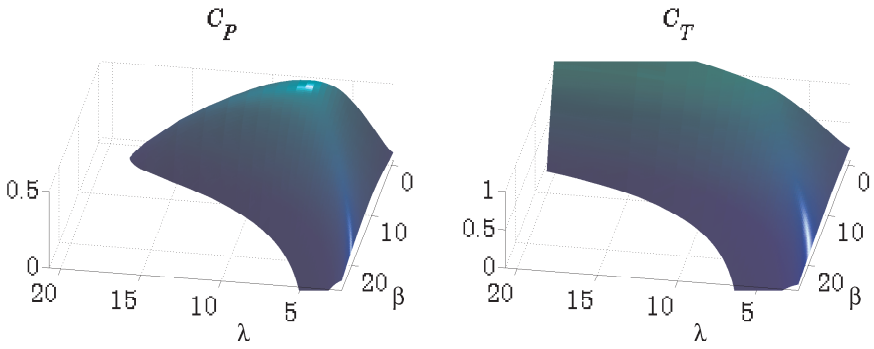

Figure 2.6: The power coefficient C_P (left) and the thrust coefficient C_T (right) as functions of the tip speed-ratio λ and pitch angle β.

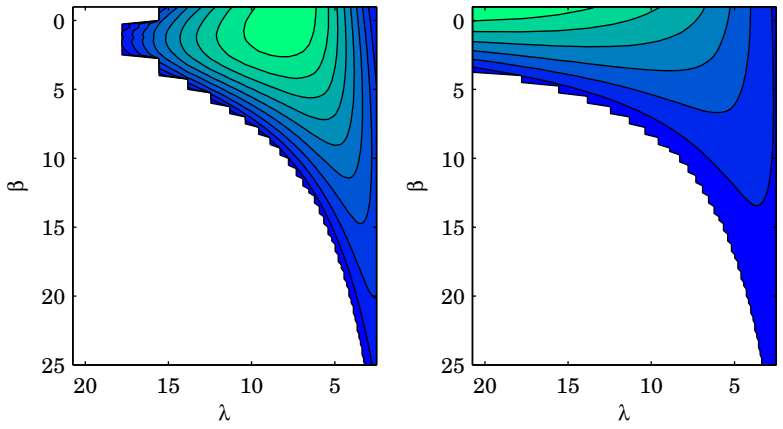

Figure 2.7: The power coefficient C_P (left) and the thrust coefficient C_T (right) as functions of the tip speed-ratio λ and pitch angle β.

- A viscious damper with viscosity constant B_r representing the bearings in the low-speed part of the drive train (before the gear box).
- A massless, visciously damped rotational spring with spring constant K_d and visciousity B_d. The spring deformation θ, measured in radians, represents the deformation of the low-speed shaft.
- A gearbox with ratio N_g.
- A viscious damper with viscosity constant B_g representing the bearings in the high-speed part of the drive train (after the gear box).
- The combined rotational moment of inertia I_g of the gearbox, high-speed shaft, and the generator.

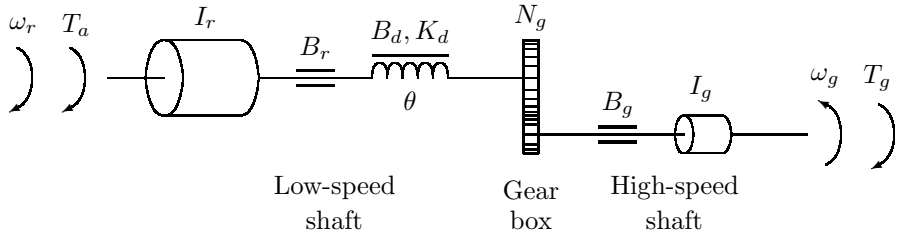

Figure 2.8: Mechanical equivalent for the drive train.

The combined differential equations describing this system are:

$$I_r\dot{\omega}_r = T_a - K_d\theta - B_d\left(\omega_r - \frac{\omega_g}{N_g}\right) - B_r\omega_r \tag{2.7a}$$

$$I_g\dot{\omega}_g = -T_g + \frac{K_d}{N_g}\theta + \frac{B_d}{N_g}\left(\omega_r - \frac{\omega_g}{N_g}\right) - B_g\omega_g \tag{2.7b}$$

$$\dot{\theta} = \omega_r - \frac{\omega_g}{N_g}. \tag{2.7c}$$

Here, the aerodynamic torque T_a and the generator torque T_g are the inputs to the system, and the resulting rotational speeds of the rotor and the generator, denoted ω_r and ω_g, are outputs, as depicted in figure 2.3.

As (2.7) constitutes a system of coupled, linear differential equations, they readily lend themselves to a linear state-space representation as follows:

$$\begin{pmatrix}\dot{\omega}_r\\ \dot{\omega}_g\\ \dot{\theta}\end{pmatrix} = \begin{pmatrix}\frac{-B_r-B_d}{I_r} & \frac{B_d}{N_gI_r} & -\frac{K_d}{I_r}\\ \frac{B_d}{N_gI_g} & \frac{-B_d}{N_g^2I_g} - \frac{B_g}{I_g} & \frac{K_d}{N_gI_g}\\ 1 & -\frac{1}{N_g} & 0\end{pmatrix}\begin{pmatrix}\omega_r\\ \omega_g\\ \theta\end{pmatrix} + \begin{pmatrix}T_a\\ -T_g\\ 0\end{pmatrix}.$$

With the constants

$$I_r = 8.70 \cdot 10^6 \text{ Nms}^2 \qquad B_r = 8.50 \cdot 10^3 \text{ Nms} \qquad B_d = 2.40 \cdot 10^5 \text{ Nms}$$
$$K_d = 1.73 \cdot 10^8 \text{ Nm} \qquad N_g = 85 \qquad B_g = 6.85 \text{ Nms}$$
$$I_g = 1.50 \cdot 10^2 \text{ Nms}^2$$

the eigenvalues for the drive train system matrix are

$$-0.054988$$
$$-0.12035 \pm j13.398.$$

That is, the drive train has a poorly damped mode at app. 2.1 Hz, and a time constant of approximately 20 seconds. The time constant of 20 seconds is due to the large inertia of the rotor. These characteristics are confirmed by the drive train step response shown in figure 2.9.

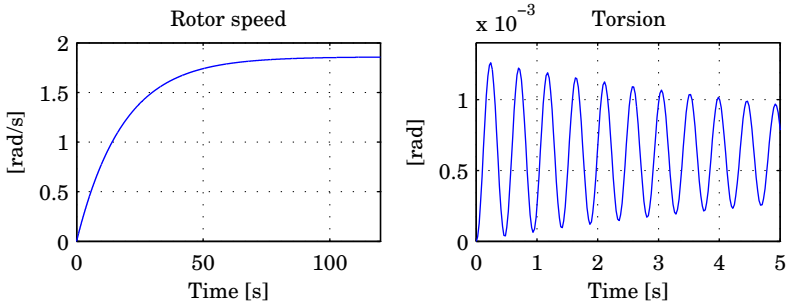

Figure 2.9: The drive train response to a 10^6 Nm step on the aerodynamic torque T_a. The 20 s time constant and the resonance frequency of app. 2.1 Hz are easily identified.

2.5 Tower

The tubular steel tower will be deflected in the fore-aft direction due to the thrust force on the rotor. A simple model approximates the deflection with a linear displacement of the nacelle, with the dynamics described by

$$m_t \ddot{z} = -K_t z - D_t \dot{z} + F_t. \tag{2.8}$$

or

$$\begin{pmatrix} \dot{z} \\ \ddot{z} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -\frac{K_t}{m_t} & -\frac{D_t}{m_t} \end{pmatrix} \begin{pmatrix} z \\ \dot{z} \end{pmatrix} + \begin{pmatrix} 0 \\ \frac{F_t}{m_t} \end{pmatrix},$$

where z, $\dot{z}$, and $\ddot{z}$ denotes position, velocity, and acceleration of the nacelle, respectively. With

$$m_t = 250 \cdot 10^3 \text{ kg} \qquad K_t = 8.88 \cdot 10^5 \text{ Nm} \qquad D_t = 296 \cdot 10^2 \text{ Ns/m}$$

the eigenvalues for the tower model becomes

$$-0.059218 \pm j1.884.$$

Thus, the tower exhibits oscillatory motion at a frequency of app. 0.3 Hz, as shown in the 10^5 N step response shown in figure 2.10.[2]

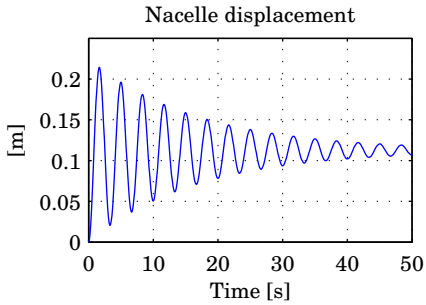

Figure 2.10: The tower has a resonance frequency of app. 0.3 Hz. Here, the response to a 10^5 Nm step on the aerodynamic thrust F_t is shown.

2.6 Generator

We will consider the generator as a device that attempts to deliver the electrical power P_e specified by the power reference signal P_{ref}. The power is controlled by adjusting the rotor current, which in turn, governs the amount of torque excerted by the generator to the high-speed shaft. Further, we will assume a loss-less generator, meaning that the electrical power equals the product between the generator speed and the generator torque:

$$P_e = \omega_g T_g. \tag{2.9}$$

Practical generators cannot change the torque instantaneously. We will model this latency by a first-order relationship between the requested generator torque

[2]The figure shows the *free* response of the tower without the rotor. This notion is important as the rotor will damp the tower motion significantly.

and the actual generator torque:

$$\dot{T}_g = \frac{1}{\tau_g}\left(T_{g,\mathrm{ref}} - T_g\right),$$

with the time constant $\tau_g = 50$ ms. As the desired torque is given by $T_{g,\mathrm{ref}} = \frac{P_{\mathrm{ref}}}{\omega_g}$, we get

$$\dot{T}_g = \frac{1}{\tau_g}\left(\frac{P_{\mathrm{ref}}}{\omega_g} - T_g\right). \tag{2.10}$$

Thus, (2.9) and (2.10) leaves the generator as a nonlinear first order MIMO system with the generator speed ω_g and the power reference P_{ref} being the inputs, and with the generator torque T_g and the produced power P_e being the outputs, as depicted in figure 2.3.

A few comments should be added to the chosen generator model. First, generator models proposed by other authors often include a time delay. When considering discrete-time models suitable for contemporary, discrete-time controller design, the inclusion of time delays in the model does not pose any problems, as time delays are readily implemented as delay states in a linear discrete-time model. This thesis concentrates on fatigue damage including stochastic wave analysis. The mathematical results needed for this analysis are all based on continuous-time descriptions. As pure time delays cannot be described by finite-dimensional, continous-time models, we choose not to include a time delay in the model.

In addition, one might claim that a more direct control of the generator could be obtained by defining the desired torque level as a controllable input to the system. Such an approach is partly protected by patent rights, cf. [MCC$^+$]. Therefore, a substantial number of commercial wind turbine control schemes rely on the indirect scheme outlined above.

2.7 Pitch actuator

The system providing blade pitch angle control consists of electrical motors, gears, and eletronic control circuits. Thus, developing a detailed model of the pitch system is exhaustive, and we will stick to the model proposed in [CO04]. That is—a first order system with a time delay. We will, though, for the same reasons as stated for the generator model, leave out the pure time delay and use the following first order relation between the pitch angle reference β_{ref} and the actual blade pitch angle β:

$$\dot{\beta} = \frac{1}{\tau_\beta}\left(\beta_{\mathrm{ref}} - \beta\right), \tag{2.11}$$

with the time constant $\tau_\beta = 120$ ms.

2.8 Neglected dynamics

The wind turbine model described in the previous sections constitutes a rather simple wind turbine model, as dynamics crucial to practical wind turbine design have been left unmodelled. One major simplification lies in the fact that the rotor has been modelled as nothing but a stiff inertia. As a consequence, blade dynamics important for the pitch bearing loads have been left out. As will be shown later, the neglect of blade dynamics necessitates a rather simplistic approach to pitch bearing fatigue. Further, neither gyroscopic effects or gravitational effects have been modelled.

Another limitation is the neglect of absolute azimuth angle. That is, the absolute, angular position of the rotor. In practical wind turbines, imperfections in rotor symmetry will cause periodic fluctuations in the mechanical loads, with a frequency equal to the rotor speed. Such effects are denoted 1P effects as they occur one time in each period in the azimuth angle.

Similarly, 3P effects arise as a result of the effective wind field experienced by the rotor not being homogenious. The effect of such inhomogenities will be periodic with a period one third of the rotor period, as there are three blades on the turbine. Well-known effects as wind shear and tower shadow are 3P effects.

Finally, note that this work does not include any model validation. As the model is very similar to the one described in [CO04], we will refer to the model validation described herein.

2.9 Model summary

Defining the state vector x, the input vector u, the output vector y, and the disturbance vector w as follows:

$$x \equiv \begin{pmatrix} \omega_r \\ \omega_g \\ \theta \\ z \\ \dot{z} \\ T_g \\ \beta \\ \tilde{v} \\ \dot{\tilde{v}} \end{pmatrix} \qquad u \equiv \begin{pmatrix} \beta_{\text{ref}} \\ P_{\text{ref}} \end{pmatrix} \qquad y \equiv \begin{pmatrix} \omega_g \\ P_e \end{pmatrix} \qquad w \equiv \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \varepsilon \end{pmatrix} \tag{2.12}$$

allows us to summarise the model as follows[3]:

$$\dot{x} = f\left(x, u, w; \bar{v}\right) \tag{2.13a}$$
$$y = g\left(x\right), \tag{2.13b}$$

with the vector-valued function f given by:

$$f\left(x, u, w; \bar{v}\right) = \begin{pmatrix} \frac{1}{I_r}\left(T_a - K_d\theta - B_d\left(\omega_r - \frac{\omega_g}{N_g}\right) - B_r\omega_r\right) \\ \frac{1}{I_g}\left(-T_g + \frac{K_d}{N_g}\theta + \frac{B_d}{N_g}\left(\omega_r - \frac{\omega_g}{N_g}\right) - B_g\omega_g\right) \\ \omega_r - \frac{\omega_g}{N_g} \\ \dot{z} \\ \frac{1}{m_t}\left(-K_t z - D_t\dot{z} + F_t\right) \\ \frac{1}{\tau_g}\left(\frac{P_{\text{ref}}}{\omega_g} - T_g\right) \\ \frac{1}{\tau_\beta}\left(\beta_{\text{ref}} - \beta\right) \\ \dot{\tilde{v}} \\ -\frac{1}{\tau_1\tau_2}\tilde{v} - \frac{\tau_1+\tau_2}{\tau_1\tau_2}\dot{\tilde{v}} + \varepsilon \end{pmatrix} \tag{2.13c}$$

where

$$T_a = \frac{P_a}{\omega_r} = \frac{1}{\omega_r}\frac{1}{2}\rho\pi R^2 v_r^3 C_P\left(\lambda, \beta\right) \tag{2.13d}$$
$$F_t = \frac{1}{2}\rho\pi R^2 v_r^2 C_T\left(\lambda, \beta\right) \tag{2.13e}$$
$$v_r = \bar{v} + \tilde{v} - \dot{z} \tag{2.13f}$$
$$\lambda = \frac{\omega_r R}{v_r}. \tag{2.13g}$$

[3]Note that x, u, and w are *variables* while the mean wind speed $\bar{v}$ is considered a *parameter*.

The output function g is given by

$$g(x) = \begin{pmatrix} \omega_g \\ \omega_g T_g \end{pmatrix}. \tag{2.13h}$$

2.10 Model linearisation

The results presented in the following chapters rely heavily on linear system models. Therefore, we will present a linearisation scheme for the model presented in section 2.9.

Let the state, input, output, and disturbance vectors be desribed as pertubed equilibrium point vectors as follows:

$$x = \bar{x} + \tilde{x} \quad , \quad u = \bar{u} + \tilde{u} \quad , \quad y = \bar{y} + \tilde{y} \quad , \quad w = \bar{w} + \tilde{w},$$

where the bar indicates the linearisation point, and the tilde indicates the deviation from the linearisation point. This gives, for the derivative:

$$\dot{x} = \dot{\bar{x}} + \dot{\tilde{x}} = f(\bar{x} + \tilde{x}, \bar{u} + \tilde{u}, \bar{w} + \tilde{w}; \bar{v}).$$

A 1st order Taylor expansion of the function $f(x, u, w; \bar{v})$ around $(\bar{x}, \bar{u}, \bar{w})$ yields

$$\dot{\bar{x}} + \dot{\tilde{x}} \approx f(\bar{x}, \bar{u}, \bar{w}; \bar{v}) + \frac{\partial f(\bar{x}, \bar{u}, \bar{w}; \bar{v})}{\partial x}\tilde{x} + \frac{\partial f(\bar{x}, \bar{u}, \bar{w}; \bar{v})}{\partial u}\tilde{u} + \frac{\partial f(\bar{x}, \bar{u}, \bar{w}; \bar{v})}{\partial w}\tilde{w}.$$

Subtracting $\dot{\bar{x}} = f(\bar{x}, \bar{u}, \bar{w}; \bar{v})$ from both sides gives the linear description

$$\dot{\tilde{x}} = A\tilde{x} + B\tilde{u} + B_w\tilde{w},$$

where the matrices A, B, and B_w are given as the Jacobians

$$A = \frac{\partial f(\bar{x}, \bar{u}, \bar{w}; \bar{v})}{\partial x} \quad , \quad B = \frac{\partial f(\bar{x}, \bar{u}, \bar{w}; \bar{v})}{\partial u} \quad , \quad B_w = \frac{\partial f(\bar{x}, \bar{u}, \bar{w}; \bar{v})}{\partial w}.$$

Following the same lines as for the state deviation vector $\tilde{x}$, the output deviation vector $\tilde{y}$ in a linearised model can be described by

$$\tilde{y} = C\tilde{x},$$

where

$$C = \frac{\partial g(\bar{x})}{\partial x}.$$

In equilibrium, $f(x, u, w; \bar{v}) = 0$. From (2.13c), we can deduce the following, rather intuitive properties of an equilibrium point:

$$\begin{aligned}
&\bar{\omega}_g = N_g \bar{\omega}_r && \dot{z} = 0 && \bar{P}_{\text{ref}} = \bar{T}_g \bar{\omega}_g = \bar{P}_e \\
&\bar{\beta} = \bar{\beta}_{\text{ref}} && \varepsilon = 0 && \bar{P}_a = \frac{1}{2}\rho\pi R^2 \bar{v}^3 C_P\left(\frac{\bar{\omega}_g R}{N_g \bar{v}}\right) \\
&\tilde{v} = 0 && \dot{\tilde{v}} = 0 && \bar{P}_e = \bar{P}_a - B_r \bar{\omega}_r^2 - B_g \bar{\omega}_g^2 \\
&\bar{v}_r = \bar{v}. &&
\end{aligned}$$

Obtaining the Jacobians is trivial but lengthy. As a result, we will only state the resulting matrices. For the system matrix A we have:

$$A = \begin{pmatrix}
\frac{T_a^{(\omega_r)} - B_d - B_r}{I_r} & \frac{1}{I_r}\frac{B_d}{N_g} & -\frac{K_d}{I_r} & 0 & \frac{T_a^{(\dot{z})}}{I_r} & 0 & \frac{T_a^{(\beta)}}{I_r} & \frac{T_a^{(\tilde{v})}}{I_r} & 0 \\
\frac{1}{I_g}\frac{B_d}{N_g} & \frac{-B_d}{I_g N_g^2} - \frac{B_g}{I_g} & \frac{1}{I_g}\frac{K_d}{N_g} & 0 & 0 & -\frac{1}{I_g} & 0 & 0 & 0 \\
1 & -\frac{1}{N_g} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
\frac{F_t^{(\omega_r)}}{m_t} & 0 & 0 & -\frac{K_t}{m_t} & \frac{F_t^{(\dot{z})} - D_t}{m_t} & \frac{F_t^{(\beta)}}{m_t} & \frac{F_t^{(\tilde{v})}}{m_t} & 0 & 0 \\
0 & -\frac{1}{\tau_g}\frac{P_{\text{ref}}}{\bar{\omega}_g^2} & 0 & 0 & 0 & -\frac{1}{\tau_g} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{\tau_\beta} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{\tau_1\tau_2} & -\frac{\tau_1+\tau_2}{\tau_1\tau_2}
\end{pmatrix}$$

where

$$\begin{aligned}
T_a^{(\omega_r)} &= \frac{\partial T_a}{\partial \omega_r} = \frac{1}{\bar{\omega}_r}\frac{1}{2}\rho\pi R^3 \bar{v}^2 \frac{\partial C_P\left(\bar{\lambda}, \bar{\beta}\right)}{\partial \lambda} - \frac{1}{\bar{\omega}_r^2}\frac{1}{2}\rho\pi R^2 \bar{v}^3 C_P\left(\bar{\lambda}, \bar{\beta}\right) \\
T_a^{(\dot{z})} &= \frac{\partial T_a}{\partial \dot{z}} = \frac{1}{2}\rho\pi R^3 \bar{v} \frac{\partial C_P\left(\bar{\lambda}, \bar{\beta}\right)}{\partial \lambda} - \frac{1}{\bar{\omega}_r}\frac{3}{2}\rho\pi R^2 \bar{v}^2 C_P\left(\bar{\lambda}, \bar{\beta}\right) \\
T_a^{(\beta)} &= \frac{\partial T_a}{\partial \beta} = \frac{1}{\bar{\omega}_r}\frac{1}{2}\rho\pi R^2 \bar{v}^3 \frac{\partial C_P\left(\bar{\lambda}, \bar{\beta}\right)}{\partial \beta} \\
T_a^{(\tilde{v})} &= \frac{\partial T_a}{\partial \tilde{v}} = \frac{1}{\bar{\omega}_r}\frac{3}{2}\rho\pi R^2 \bar{v}^2 C_P\left(\bar{\lambda}, \bar{\beta}\right) - \frac{1}{2}\rho\pi R^3 \bar{v} \frac{\partial C_P\left(\bar{\lambda}, \bar{\beta}\right)}{\partial \lambda} \\
F_t^{(\omega_r)} &= \frac{\partial F_t}{\partial \omega_r} = \frac{1}{2}\rho\pi R^3 \bar{v} \frac{\partial C_T\left(\bar{\lambda}, \bar{\beta}\right)}{\lambda} \\
F_t^{(\dot{z})} &= \frac{\partial F_t}{\partial \dot{z}} = \frac{1}{2}\rho\pi R^3 \bar{\omega}_r \frac{\partial C_T\left(\bar{\lambda}, \bar{\beta}\right)}{\partial \lambda} - \rho\pi R^2 \bar{v} C_T\left(\bar{\lambda}, \bar{\beta}\right) \\
F_t^{(\beta)} &= \frac{\partial F_t}{\partial \beta} = \frac{1}{2}\rho\pi R^2 \bar{v}^2 \frac{\partial C_T\left(\bar{\lambda}, \bar{\beta}\right)}{\partial \beta} \\
F_t^{(\tilde{v})} &= \frac{\partial F_t}{\partial \tilde{v}} = \rho\pi R^2 \bar{v} C_T\left(\bar{\lambda}, \bar{\beta}\right) - \frac{1}{2}\rho\pi R^3 \bar{\omega}_r \frac{\partial C_T\left(\bar{\lambda}, \bar{\beta}\right)}{\partial \lambda}.
\end{aligned}$$

The partial derivatives $\frac{\partial C_P}{\partial \lambda}$, $\frac{\partial C_P}{\partial \beta}$, $\frac{\partial C_T}{\partial \lambda}$, and $\frac{\partial C_T}{\partial \beta}$ are computed numerically from the C_P and C_T lookup tables.

For the input matrix B, the disturbance input matrix B_w, and the output matrix C we have:

$$B = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & \frac{1}{\tau_g \bar{\omega}_g} \\ \frac{1}{\tau_\beta} & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \quad , \quad B_w = I \quad , \quad C = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \bar{T}_g & 0 & 0 & 0 & \bar{\omega}_g & 0 & 0 & 0 \end{pmatrix}.$$

Thus, the linearised model is described by the following state-space model[4]:

$$\dot{\tilde{x}} = A\tilde{x} + B\tilde{u} + w \tag{2.14a}$$

$$\tilde{y} = C\tilde{x} \tag{2.14b}$$

with w being a white noise process with the intensity matrix R_1:

$$R_1 = \begin{pmatrix} 0 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & 0 \\ 0 & \cdots & 0 & \left(\frac{k}{\tau_1+\tau_2}\right)^2 \end{pmatrix}. \tag{2.15}$$

[4]In the linearisation point, $\varepsilon = 0$. Thus, $\tilde{w} = w$ and we omit the tilde.

Chapter 3

Wind turbine control

In this chapter the basics of wind turbine control are outlined.

First, the wind turbine is defined as a control object. This includes a discussion on optimal equilibrium points of the model, resulting in the definition of operation modes.

Next, the design and implementation of a simple hybrid wind turbine controller is described. The hybrid controller is a four-mode controller that includes three different PI-controllers along with a nonlinear control scheme referred to as a P, ω-controller.

Finally, an LQ control setup with integral action is described, and a controller tuning example is given, demonstrating some fundamental tradeoffs inherent to wind turbine controller design.

3.1 Existing works

Several authors have described PI control schemes for wind turbine control, including gain scheduled setups. Also more general MIMO control strategies such as LQG controllers have been proposed. Most works concentrate on reducing

the loads on the turbine using techniques such as split-pitch control and active drive train damping.

Even though the resulting controllers do lower the fatigue damage inflicted on the turbine, none of the design paradigms brought into play aim directly at minimizing fatigue damage. Techniques such as LQG design rely upon a quadratic performance measure, which can be related to the variance of the load. Even though a decrease in variance usually would decrease the fatigue damage, this is not the case in general, and as such, these methods cannot be categorized as directly aiming at fatigue damage reduction.

As a result, most works concludes that considerable effort lies in finding the right tuning of the controllers, as the trade-offs providing a desirable solution in terms of fatigue loads are not obvious.

Wind turbine operation is subject to a number of constraints on the inputs and outputs. An elegant handling of these constraints was proposed in [CO04], where these constraints were combined with a quadratic cost to form a gain-scheduled model predictive control setup.

3.2 The wind turbine as a control object

The model described in chapter 2 leaves a dynamic system with three inputs: wind speed v, pitch angle reference signal β_{ref}, and power reference P_{ref}.

Pitch angle reference and power reference are considered controllable inputs and the wind speed is an uncontrollable disturbance. We will define the generator speed ω_g and the produced power P_e as the primary output y of interest. The information available to the controller, y_m, will be a function of the system states x as well as the current control signal u. Thus, the general control setup can be depicted as in figure 3.1.

3.2.1 Optimal equilibrium points

As the very purpose of a wind turbine is the one of generating power, one would expect the wind turbine to be operated at constant tip speed-ratio $\lambda = \lambda^*$ and constant blade pitch angle $\beta = \beta^*$ at all times in order to obtain maximum power efficiency, $C_P = C_P^*$.

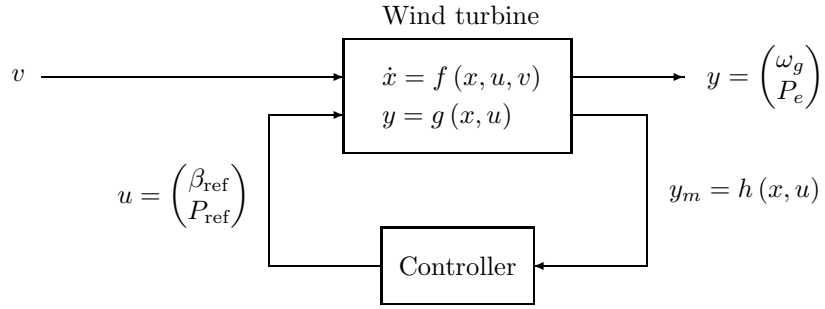

Figure 3.1: The general wind turbine control setup.

There are, however, practical limitations that prevent such operation. Generator speed ω_g has to be kept within a certain operating range, and produced power P_e should not exceed nominal power, P_0. Thus, the problem of defining the optimal equilibrium point $\langle \bar{\omega}_g^*, \bar{\beta}^* \rangle$ of operation for a given mean wind speed $\bar{v}$ can be formulated as follows:

$$\langle \bar{\omega}_g^*, \bar{\beta}^* \rangle = \underset{\langle \bar{\omega}_g^*, \bar{\beta}^* \rangle}{\operatorname{argmax}} \bar{P}_e \tag{3.1a}$$

subject to

$$c_1 : \quad \bar{\omega}_g > \omega_{g,\min} \tag{3.1b}$$
$$c_2 : \quad \bar{\omega}_g < \omega_{g,\max} \tag{3.1c}$$
$$c_3 : \quad \bar{P}_e \leq P_0 \tag{3.1d}$$

where

$$\bar{P}_e = \frac{1}{2}\rho\pi R^2 \bar{v}^3 C_P \left(\frac{\bar{\omega}_g R}{N_g \bar{v}}, \bar{\beta} \right) - \bar{\omega}_g^2 \left(\frac{B_r}{N_g^2} + B_g \right). \tag{3.1e}$$

As the objective $\bar{P}_e$ is a function of the wind speed $\bar{v}$, the optimal operating point is also a function of the wind speed.

Solving the optimization problem (3.1) results in four different types of solutions, determined by the set of active constraints at the solution. We will designate the four types as *operation modes* as summarised in table 3.1.

In figure 3.2 the optimal operating point characteristics (generator speed, pitch angle, and electrical power) are plotted a functions of the mean wind speed.

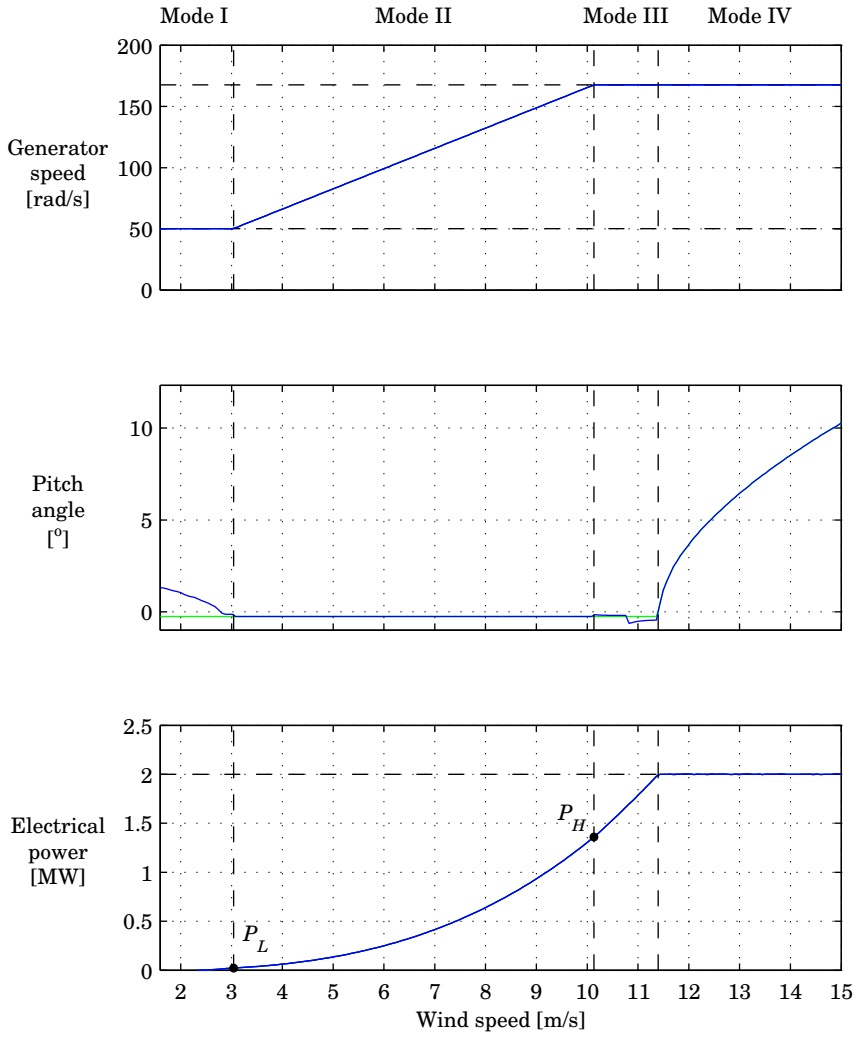

Figure 3.2: Optimal equilibrium points (blue) and suboptimal equilibrium points (green)—the latter allowing a SISO approach.

Mode	Active constraints	Properties
I	c_1	$\omega_g = \omega_{g,\min}$, $\beta \neq \beta^*$
II		$\omega_g \propto v_r$, $\beta = \beta^*$
III	c_2	$\omega_g = \omega_{g,\max}$, $\beta \neq \beta^*$
IV	c_2, c_3	$\omega_g = \omega_{g,\max}$, $\beta \neq \beta^*$, $P_e = P_0$

Table 3.1: Characteristics of the four operating modes. See figure 3.2.

3.3 A PI approach

In order to investigate some of the fundamental implications of wind turbine control, we will consider a hybrid control scheme using mode-dependent PI control strategies.

The controller design will be loosely based on the design suggested in [HHL+05], which is based on the following two concepts:

- The turbine is treated as a SISO system with the control input being either the power reference or the pitch reference, depending on operation mode.
- The turbine is modelled as a first-order system, allowing for a pole placement design for the second-order system resulting from applying the first-order PI-controller.

3.3.1 The SISO approximation

Figure 3.2 shows that the pitch angle β is held almost constant at β^* in mode I,II, and III. Furthermore, as the power is held constant in mode IV, it is tempting to treat the wind turbine—which inherently is a 2×2 MIMO system—as a SISO system with the control strategy determined by operating mode as depicted in figure 3.3.

Mode I Generator speed is held constant at $\omega_g = \omega_{g,\min}$ by a PI-controller acting on the power reference. Pitch held constant at $\beta = \beta^*$.

Mode II Power reference adjusted according to the nonlinear relationship between generator speed and power during C_P^*-operation (stationary condition):

$$P_{\text{ref}} = P_e = P_a - B_r\omega_r^2 - B_g\omega_g^2 = \frac{1}{2}\rho\pi R^5 C_P^* \frac{1}{\left(\lambda^* N_g\right)^3}\omega_g^3 - \left(\frac{B_r}{N_g^2} + B_g\right)\omega_g^2.$$

Mode III Generator speed is held constant at $\omega_g = \omega_{g,\max}$ by a PI-controller acting on the power reference. Pitch held constant at $\beta = \beta^*$. Basically the same as mode I, but with different setpoint.

Mode IV Generator speed is held constant at $\omega_g = \omega_{g,\max}$ by a PI-controller acting on the pitch angle reference.

If the current wind speed is known by the controller, the switching conditions for toggling between operation modes would readily follow from figure 3.2.

We will assume, though, that the wind speed is not known by the controller. This means that the mechanism swithing between the control modes will use the operating point characteristic ω_g, P_e, and β for mode selection as depicted in figure 3.4. Note how the power levels P_L and P_H marked in figure 3.2 are used as transition conditions.

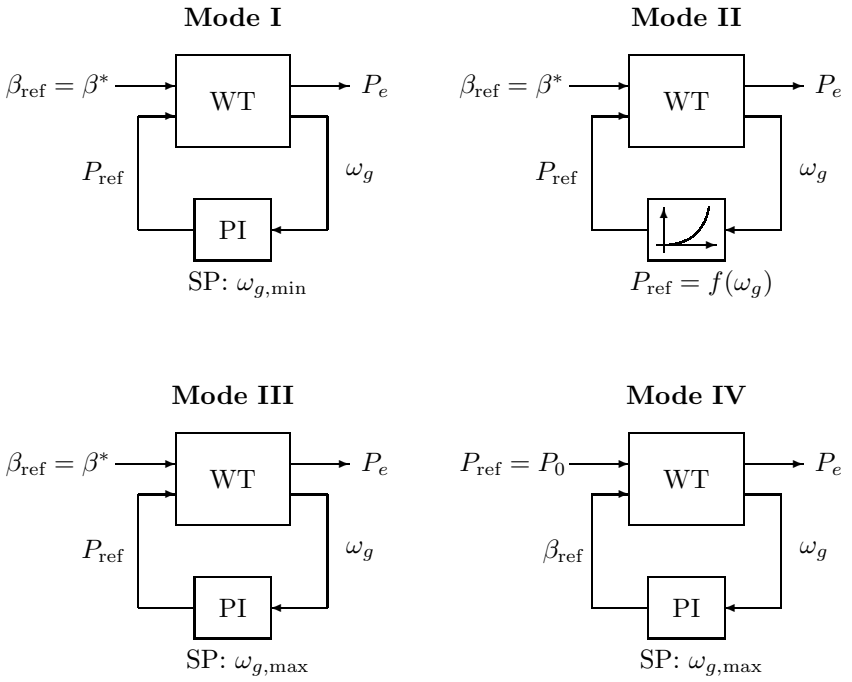

Figure 3.3: The four different control schemes in the hybrid controller. SP denotes controller setpoint.

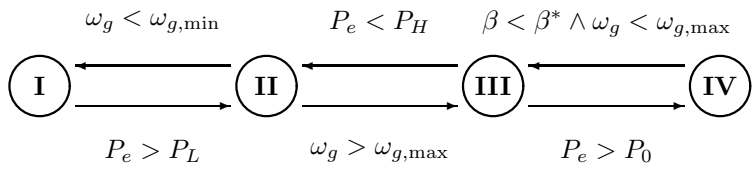

Figure 3.4: Mode transitions in the hybrid controller. The circles represent the modes and the arrow captions denote the condition for the mode transition indicated by the arrow.

3.3.2 Controller design

In order to make the desired pole placement design, we need to obtain a first-order wind turbine model.

A standard method for approximating (2.14) with a first order model is to transform the model into a balanced realisation with the observability and controllability matrices being identical diagonal matrices with the Hankel singular values occuring in descending order in the diagonal. This transformation allows for a direct measure of the influence each (transformed) state has on the input-ouput relationship. Neglecting all states but the first in the transformed model leaves a first order system.

Applying off-the-shelf MATLAB routines for the model reduction described above yields unsatisfactory results, though. Due to the model (2.14) being unstable in a certain region of operation (first part of the mode IV region), the reduced system will have system parameters that are discontinuous when plotted as a function of the mean wind speed. This is mainly due to the fact that the standard model reduction routines will leave unstable parts of the original model unaltered, thus introducing a discontinuity when the original system model changes from being unstable to being stable.

As the gain schedule developed in section 3.3.2.1 will rely on a the model parameters being smooth functions of the wind speed, we will, instead, develop a first-order model based on the same first-principles as was used in chapter 2, with the crude assumption of a rigid drive train and a rigid tower as well as ideal pitch actuators and generator. The details of this diminished modelling are found in appendix A, where also the accompaning pole placement PI-controller design is outlined.

In the design, the s-plane poles are placed at $-0.5 \pm j0.5$, resembling the design suggested in [HHL$^+$05].

3.3.2.1 Gain scheduling

As the linearised wind turbine model varies with the operating point, the controller gains should be adjusted according to the present operating point. In figure 3.5, the optimal gains are plotted as functions of the wind speed. The plot shows that controller gains are practically constant for the mode I and mode III controllers, while the gains of the mode IV controller varies significantly. Therefore, a gain scheduling scheme is applied to this controller.

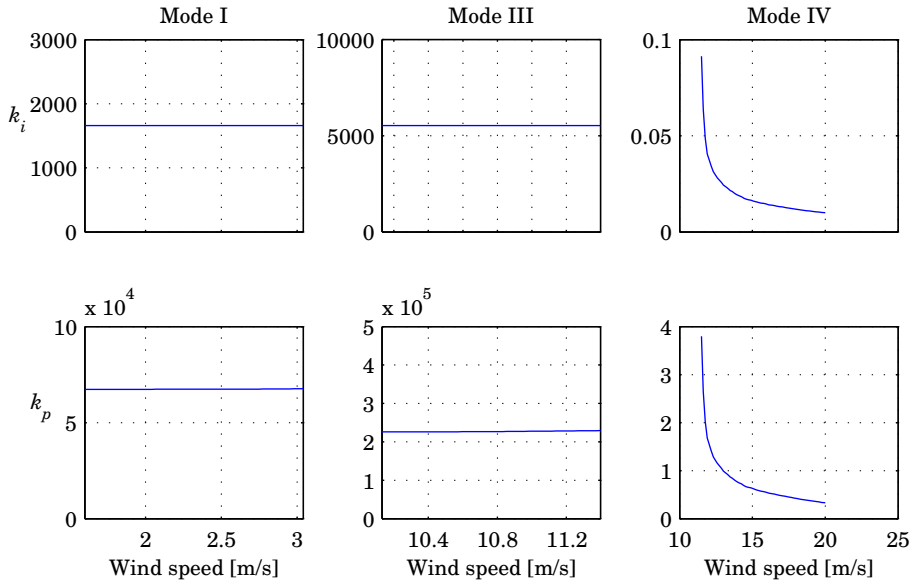

Figure 3.5: Optimal PI controller gains k_p and k_i as functions of wind speed. Gains are practically constant in modes I and III.

As the wind speed is considered unknown, we will use the pitch angle as an indicator of the current wind speed. This approach is justified by the integral action in the PI controller, as this ensures asymptotic convergence towards the desired operating point for a given wind speed. This implies that, in stationarity, the mode IV relationship between wind speed and pitch angle depicted in figure 3.2 will be reached, effectively leaving the the pitch angle as a wind speed estimator.

In figure 3.6, the controller gains are plotted as a function of the pitch angle.

We choose to approximate the gains with the functional relationships

$$k_x(\beta) = \frac{p_x}{\beta + q_x}.$$

A least squares-fit results in the models

$$k_i(\beta) = \frac{0.17595}{\beta + 0.75768} \tag{3.2a}$$

and

$$k_p(\beta) = \frac{6.824}{\beta + 0.6016}. \tag{3.2b}$$

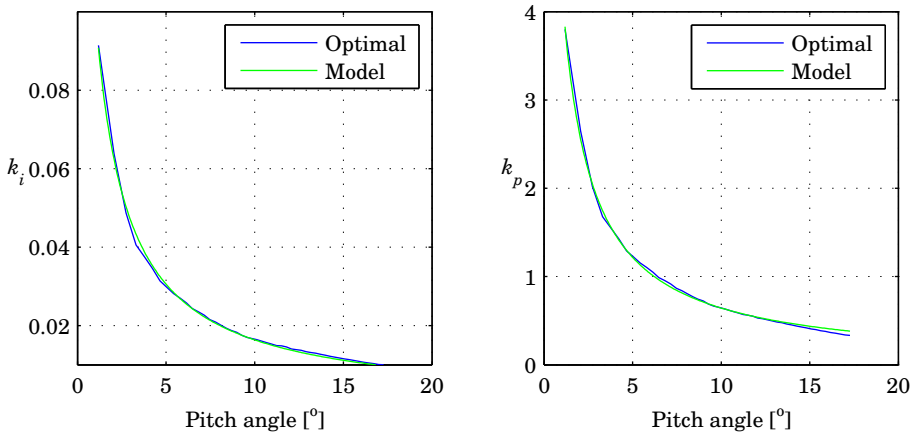

Figure 3.6: Optimal controller gains k_i and k_p for mode IV PI controller as a function of pitch angle β. Gains are approximated by the models (3.2a) and (3.2b).

A MATLAB-implementation of the discrete-time hybrid controller function is seen in appendix A.4. Note that bumpless transfer between modes is assured by adjusting the integrator state when the mode changes in such a way that the control variable will not change abruptly.

3.3.3 Controller verification

Initial simulations of the controller immediately reveals that the controller designed as described in the previous sections fails to stabilize the system. Analysis of the closed-loop system reveals unstable eigenfrequencies close to the drive

train resonance frequency at ~ 2.1 Hz. Therefore, the hybrid controller is extended to include a bandstop-filtering of the generator speed signal before it is fed to the controller, as depicted in figure 3.7. This solution is also found in other turbine control schemes to mitigate the effects of drive train vibrations.

Figure 3.7: The hybrid controller setup is stabilised by bandstop-filtering the generator speed signal before the controllers.

Having stabilised the closed-loop system by the bandstop-filter, we are ready to investigate the performance of the hybrid controller. Figure 3.8 shows the closed-loop system behaviour when operated in the mode II/III/IV transition region. We note the following properties:

- The produced power is limited at nominal power (2 MW) during mode IV operation.
- Pitch angle is held constant at $\beta = \beta^* = -0.6°$ in modes II and III.
- Pitch angle rate of change is kept within $\pm 10°/\text{s}$.
- Bumpless transfer between operation modes is observed.

More simulation results are found in appendix A.3, illustrating the following scenarios:

Quasi-stationary operation In order to investigate the steady-state properties of the closed-loop system, the system is excerted to a slowly varying wind speed, resulting in what could be denoted "quasi-stationary" operation. The linear growth of the wind speed allows for comparison with the optimal equilibrium points shown in figure 3.2.

Deterministic mode transition simulation Demonstrates stability and bumpless transfer around mode transitions.

Deterministic mode IV step-responses Demonstrates that, due to the gain scheduling, the dynamic properties of the closed-loop system are unaffected by the changing system dynamics.

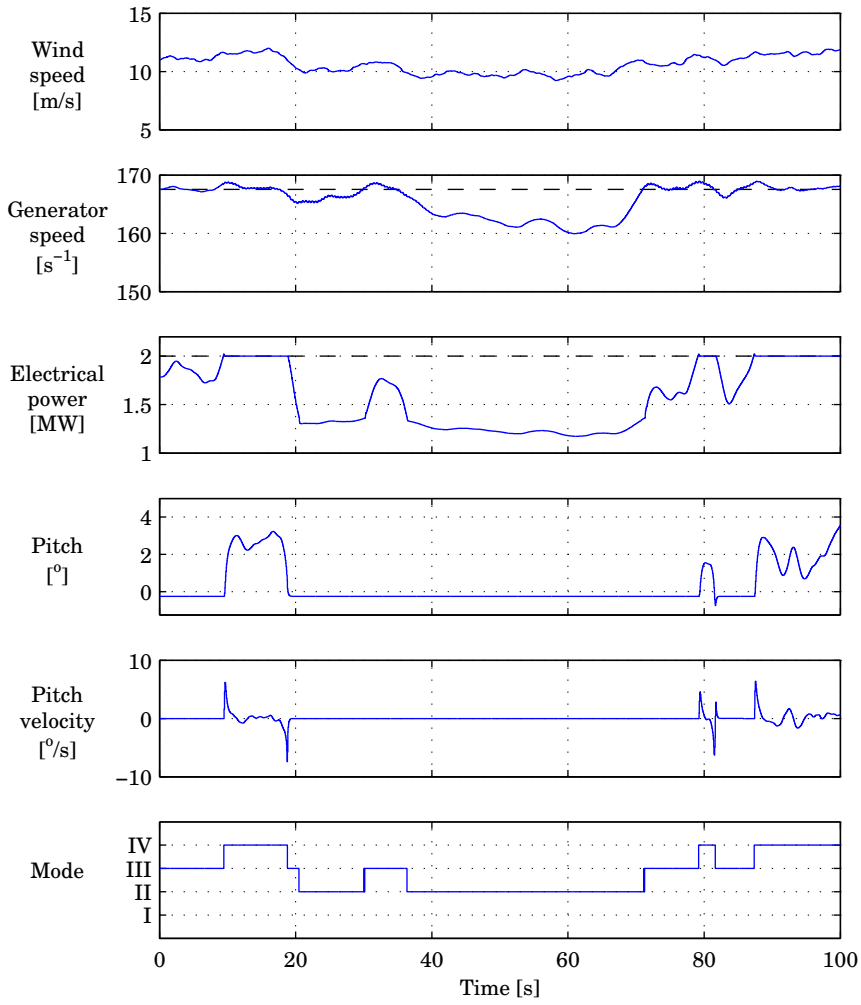

Figure 3.8: Non-linear simulation of the hybrid controller.

3.4 An LQI appoach

Attention is now turned towards the class of state feedback controllers. This class of controllers allows for arbitrary pole placement, and is inherently suited for MIMO systems. A particular useful design technique is the LQ-design that—besides from providing stable closed-loop systems—allows for an intuitively appealing tuning procedure. Adding integral states to the setup ensures correct reference following despite the model uncertainties inherently present in a linearised model of a physical system.

3.4.1 The LQI setup

Consider the linearised state-space model described in terms of the deviation variables $\tilde{x}$, $\tilde{u}$, and $\tilde{y}$:

$$\begin{aligned}\dot{\tilde{x}} &= A\tilde{x} + B\tilde{u} + w\\ \tilde{y} &= C\tilde{x},\end{aligned}$$

where w is a white noise process with intensity matrix R_1. Introducing the integral state vector x_i:

$$x_i \equiv \begin{pmatrix}\omega_g^i & P_e^i\end{pmatrix}^T$$

with

$$\dot{x}_i = \tilde{y} = C\tilde{x},$$

we define the state vector $\tilde{x}_a$ for the augmented system:

$$\tilde{x}_a \equiv \begin{pmatrix}\tilde{x}\\ x_i\end{pmatrix}. \tag{3.3}$$

Combining these equations leads to the augmented system description

$$\begin{aligned}\dot{\tilde{x}}_a &= \begin{pmatrix}\dot{\tilde{x}}\\ \dot{x}_i\end{pmatrix} = \begin{pmatrix}A & 0\\ C & 0\end{pmatrix}\begin{pmatrix}\tilde{x}\\ x_i\end{pmatrix} + \begin{pmatrix}B\\ 0\end{pmatrix}\tilde{u} + \begin{pmatrix}w\\ 0\end{pmatrix}\\ \tilde{y} &= \begin{pmatrix}C & 0\end{pmatrix}\begin{pmatrix}\tilde{x}\\ x_i\end{pmatrix}\end{aligned}$$

or

$$\dot{\tilde{x}}_a = A_a\tilde{x}_a + B_a\tilde{u} + w_a \tag{3.4a}$$

$$\tilde{y} = C_a\tilde{x}_a. \tag{3.4b}$$

where w_a is a white noise process with intensity matrix $R_{1,a}$:

$$R_{1,a} = \begin{pmatrix} R_1 & 0 \\ 0 & 0 \end{pmatrix}$$

Now, consider the cost J:

$$J = E\left\{ \int \epsilon^T Q \epsilon dt \right\}, \tag{3.5}$$

where the error vector ϵ is defined as follows[1]:

$$\epsilon = \begin{pmatrix} \tilde{\omega}_g & \tilde{P}_e & \tilde{\theta} & \tilde{z} & \tilde{\beta} & \tilde{\omega}_g^i & \tilde{P}_e^i & \tilde{\beta}_{\text{ref}} & \tilde{P}_{\text{ref}} \end{pmatrix}^T. \tag{3.6}$$

This cost can be interpreted as a weighted sum of the variances of the elements in ϵ. The cost can—assuming a diagonal Q matrix—be rewritten as

$$J = \int \begin{pmatrix} \tilde{y}^T & \tilde{x}^T & \tilde{x}_i^T & \tilde{u}^T \end{pmatrix} \begin{pmatrix} Q_{\tilde{y}} & 0 & 0 & 0 \\ 0 & Q_{\tilde{x}} & 0 & 0 \\ 0 & 0 & Q_i & 0 \\ 0 & 0 & 0 & Q_{\tilde{u}} \end{pmatrix} \begin{pmatrix} \tilde{y} \\ \tilde{x} \\ \tilde{x}_i \\ \tilde{u} \end{pmatrix} dt \tag{3.7}$$

where

$$Q_{\tilde{y}} \equiv \begin{pmatrix} Q_{\tilde{\omega}_g} & 0 \\ 0 & Q_{\tilde{P}_e} \end{pmatrix} \quad , \quad Q_i \equiv \begin{pmatrix} Q_{\tilde{\omega}_g^i} & 0 \\ 0 & Q_{\tilde{P}_e^i} \end{pmatrix} \quad , \quad Q_{\tilde{u}} \equiv \begin{pmatrix} Q_{\tilde{\beta}_{\text{ref}}} & 0 \\ 0 & Q_{\tilde{P}_{\text{ref}}} \end{pmatrix}$$

and

$$Q_{\tilde{x}} \equiv \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & Q_{\tilde{\theta}} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & Q_{\tilde{z}} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & Q_{\tilde{\beta}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Now, (3.3) and (3.4b) gives

$$J = \int \begin{pmatrix} \tilde{x}_a^T & \tilde{u}^T \end{pmatrix} \begin{pmatrix} C_a^T Q_{\tilde{y}} C_a + Q_{\tilde{a}} & 0 \\ 0 & Q_{\tilde{u}} \end{pmatrix} \begin{pmatrix} \tilde{x}_a \\ \tilde{u} \end{pmatrix} dt$$

where

$$Q_{\tilde{a}} \equiv \begin{pmatrix} Q_{\tilde{x}} & 0 \\ 0 & Q_i \end{pmatrix}.$$

[1]Note that the entries $\tilde{\omega}_g^i$ and $\tilde{P}_e^i$ are the time integrals of the deviation variables $\tilde{\omega}_g$ and $\tilde{P}_e$

Finally, we can write

$$J = \int \left\{ \tilde{x}_a^T \underbrace{\left(C_a^T Q_{\tilde{y}} C_a + Q_{\tilde{a}} \right)}_{Q_1} \tilde{x}_a + \tilde{u}^T \underbrace{Q_{\tilde{u}}}_{Q_2} \tilde{u} \right\} dt. \tag{3.8}$$

It can be shown that the control signal $\tilde{u}$ that minimizes (3.8) subject to (3.4a) is given by the control law $\tilde{u} = -L\tilde{x}_a$.

A necessary condition for this to be true, though, is that the pair $\langle A_a, B_a \rangle$ is stabilisable. This means that all uncontrollable states in the system must be stable. Investigation of the system (3.4) reveals—not surprisingly—that the wind model states $\tilde{v}$ and $\dot{\tilde{v}}$ are not controllable. They are stable, though, so the system retains stabilisability[2].

Given Q_1 and Q_2, the MATLAB function `lqr` computes the gain matrix L, leaving Q_1 and Q_2 as the design parameters or *tuning knobs* in the LQI setup.

3.4.2 LQI controller design

The application of LQI design provides a consistent design procedure that ensures stability for the linearised closed-loop system model. It should be stressed, however, that the inherent nonlinearities of the wind turbine model and the operating point dependent operation modes treated in section 3.3 still apply. That is, a mechanism for controller gain adjustment should be implemented, and measures should be taken to ensure bump-less transfer between operation modes. The implementation of such a scheme is a bit more complicated in the LQI-case than for the PI hybrid controller, but would be based on the same principles.

In this treatment an LQI design for only one single mode IV operation point is considered.

3.4.3 Controller tuning—baseline controller

As mentioned above, the weights in the LQ design represent a weighting of the variances. A common starting point for the weights when doing LQ design is to use weights equal to the inverse of the square of the variable's nominal value. That correponds to a weighting, where the *variabilities* of the variables are weighted equally.

[2]The filter (2.2) is stable.

Thus, we have:

$$Q_{\tilde{y}} = \begin{pmatrix} \frac{1}{\bar{\omega}_g^2} & 0 \\ 0 & \frac{1}{\bar{P}_e^2} \end{pmatrix} \quad , \quad Q_i = \begin{pmatrix} \frac{1}{\bar{\omega}_g^2} & 0 \\ 0 & \frac{1}{\bar{P}_e^2} \end{pmatrix} \quad , \quad Q_{\tilde{u}} \equiv \begin{pmatrix} \frac{1}{\beta_{\text{ref}}^2} & 0 \\ 0 & \frac{1}{\bar{P}_{\text{ref}}^2} \end{pmatrix}$$

and

$$Q_{\tilde{x}} \equiv \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{\theta^2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{\bar{z}^2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{\beta^2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Note that the weights of the integral states, found in Q_i, are defined in terms of the variables the integrate, and not by the inverse of their nominal values (which are zero!).

We will denote the LQI state feedback controller designed from this textbook starting point as the *baseline* state feedback controller, with the gain matrix $L = L_0$.

Figure 3.9 shows a simulation of the baseline controller applied to the wind turbine model linearised around the mean wind speed $\bar{v} = 16$ m/s.

3.4.3.1 Controller tuning—tower damping

Now, imagine that the tower deflection represented by the nacelle position in figure 3.9 is judged to inflict too much damage on the tower structure. In that case, a different tuning of the controller might reduce the tower deflection.

As the nacelle position z exists as a term in the LQI cost function, we increase the weight $Q_{\tilde{z}}$ in the $Q_{\tilde{x}}$ matrix by a factor one hundred and recompute the controller gain matrix L. The result is depicted by the green lines in figure 3.9.

The plots show that the tower motion has been reduced slightly, as was the purpose of the redesign. The figure also shows, however, that the tower damping has a cost in terms of substantially increased low-speed shaft deformations as well as a notably higher pitch angle velocity. As will become apparent later, the pitch angle velocity is governing the fatigue damage rate of the pitch bearings.

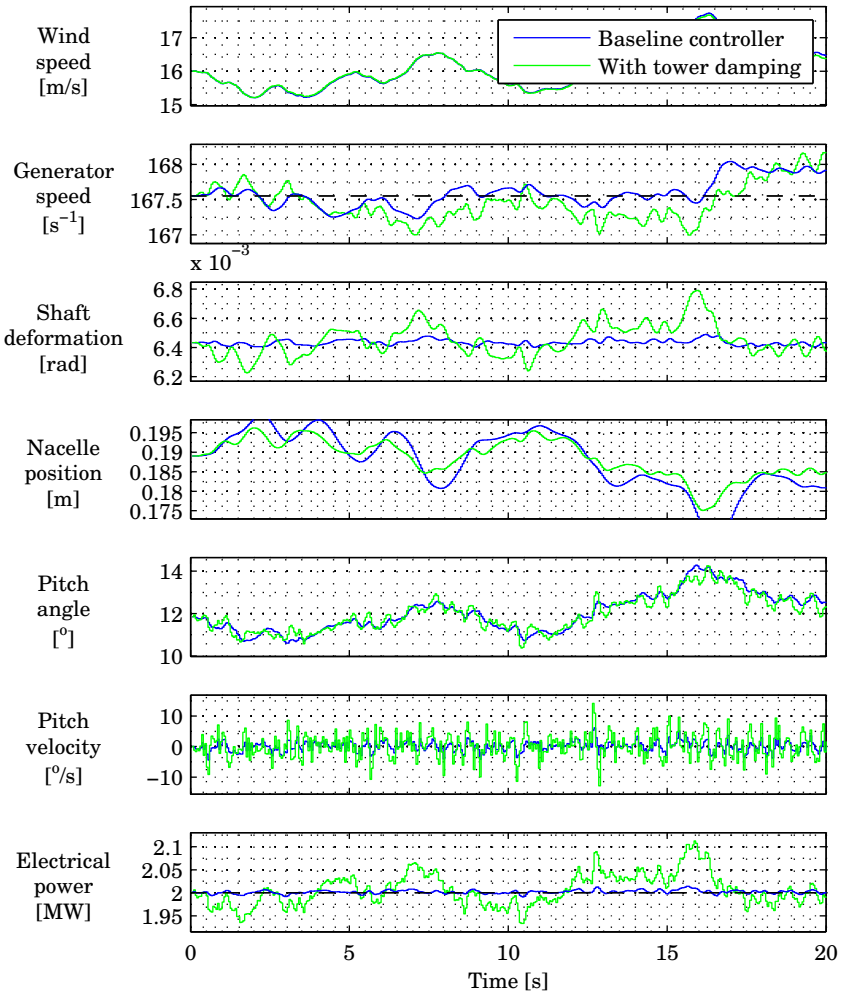

Figure 3.9: Linear simulation of the LQI controller. Blue: the baseline controller as defined in section 3.4.3. Green: design weights adjusted to lower tower deflection.

A substantial increase in the variability of the produced power is also a result of the tower damping controller design.

3.5 Conclusion

Based on a simple wind turbine model a SISO control strategy with PI-controllers has been applied. This strategy led to an unstable system, with unstable poles at the drive train resonance frequency. The band-stop filter solution applied to the stability issue is commonly used in wind turbine controllers [HHL+05, BSJB01], and as such, there is nothing suspicious about applying the solution presented above. It should be stressed, though, that the inclusion of the band-stop filter is not the result of any standard control strategy being applied to a standard control problem. It is a qualified ad-hoc solution to a specific problem.

The more general LQI state-feedback controller provided a means of designing a stabilising controller. This allowed for a tuning procedure where the tower deflection was reduced by directly adjustment of the weight related to this state variable.

It should be stressed that the LQ design procedure provides a controller that is optimal *in the sense described by the designer through the cost function.* That is, the designer still must adjust the weights by hand to obtain a controller that is optimal in the global sense as outlined in chapter 1.

Chapter 4

Fatigue

Fatigue loads is an important issue when designing structures exposed to long-term vibrations and fluctuating loads, as is the case for wind turbines. This chapter outlines the basics of fatigue damage analysis.

4.1 Introduction

When designing mechanical and structural components, two load contributions are of special importance—extreme loads and fatigue loads. Extreme loads can be thought of as loads that will cause the structure to fail due to the stress exceeding the yield strength for the material. Thus, the design goal is to avoid—by a proper low probability—that the material is exposed to such excessive loads.

Fatigue loads can be thought of as the loss of strength that a material experience when subjected to a cyclic stress history. The process often starts at the surface of a material where small imperfections such as scratches cause stress concentrations. As the crack size grows, stress concentrations grow accordingly, causing the cracks to grow further. This process eventually leads to rupture of the material.

It is important to note that fatigue occurs even when the applied loads are far below the material's elastic limit. This implies that not only the extreme load values are important; the small-amplitude time history of the load affects component reliability due to fatigue damage.

4.2 The SN-curve and Palmgren-Miner's damage rule

In order to describe a material's ability to withstand cyclic stress histories, an SN curve is often used. An SN curve can be constructed by applying a constant-amplitude cyclic stress to a test specimen and count the number of load cycles until rupture occurs. Repeating the test for a number of stress ranges allows the stress range to be plotted against the number of cycles that the material can withstand.

An often-used model for the SN curve is

$$s^k N = K, \tag{4.1}$$

where s is the stress range (twice the amplitude in a sinusoidal stress history) and N is the lifetime in cycles. The quantities K and k are material properties, with k being denoted the *Wöhler-coefficient*.

An SN-curve for a material with $k = 4$ and $K = 10^{32}$ is depicted in figure 4.1. The SN-curve shows that the (fictious) material can withstand 10^4 stress cycles of range 10^7 Pa, or 10^8 cycles with range 10^6 Pa. The curve also shows the effect of the Wöhler-coefficient k, as the slope in the double-logarithmic plot equals $-1/k$. In the example, $k = 4$, which means that an increase in stress range by a factor of 10 decreases the lifetime in cycles by a factor of 10^4.

An interesting feature of the SN curve based characterisation of a material's fatigue properties is the lack of time/frequency dependence. That is, it makes no difference if the stress cycles are applied rapidly or slowly—only the number of cycles and their ranges are important [SBD05].

Now, we will define the damage D_i imposed by a stress cycle with range s_i as

$$D_i \equiv \frac{1}{N_i} = \frac{1}{K} s_i^k. \tag{4.2}$$

Further, we introduce a linear damage accumulation rule known as Palmgren-Miner's damage rule, which states that the total damage D imposed by a given

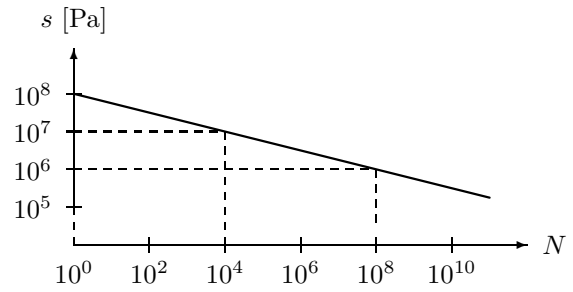

Figure 4.1: An example SN-curve. Note the double-logarithmic scale.

stress history is found as the sum of the damages imposed by the individual M cycles in the total stress history:

$$D = \sum_{i=1}^{M} \frac{1}{K} s_i^k. \tag{4.3}$$

Consider the example material described above being subjected to a stress history of $M = 10^4$ cycles, each with range $s_i = 10^7$ Pa. The total damage is thus found as

$$D = \sum_{i=1}^{M} \frac{1}{K} s_i^k = \sum_{i=1}^{10^4} \frac{1}{10^{32}} \left(10^7\right)^4 = 1.$$

This result constitutes an example of a convenient property of the damage definition (4.2): *Fatigue failure occurs when the total damage exceeds unity.*

One should be aware that the fatigue model presented here neglects the effect of the mean stress level, which in some works is accounted for by adding a mean stress correction factor to the SN curve. All works cited in this work neglects the mean stress level in their treatments of the subject.

Finally, it should be noted that due to the inherent inhomogenities in materials, fatigue is a stochastic phenomenon, which, in turn, leaves the constant K as being a stochastic variable, with $1/K$ often taken to be lognormally distributed [Gro00]. The effect of this is that the material characterisation should be expanded to include the variance of the parameter K. In this treatment, however, the effect of the material property K being a stochastic quantity will not be considered.

4.3 Rainflow counting

The previous section described a fatigue damage estimate as the combined damage from a number of stress cycles. The interpretation of a cycle is apparent when it comes to constant-amplitude, cyclic stress histories like a sine wave, for instance. In real-life applications, though, stress histories exhibit complex waveforms with the meaning of a cycle not being obvious.

A solution to this problem is to convert a given, arbitrary waveform into a number of *equivalent stress ranges*. That is, a number of stress ranges that will cause the same amount of fatigue damage as the original waveform.

Numerous algorithms for carrying out this conversion has been proposed, but, in general, the so-called *rainflow-counting* method is considered as being the superior one [Ryc93]. Several descriptions of the rainflow-counting method exist, spanning from the pagoda-roof explanation originating from Endo's original 1969 work, to the definition given in [Ryc87], which is equivalent to the following definition:

Rainflow cycle For each local maximum M_j in the stress history, identify the continuous regions to the left, and to the right of M_j where all stress values are below M_j. Next, find the minimum stress value for each of the two regions, denoted m_j^- and m_j^+ and define $m_j^{\text{rfc}} = \max\left\{m_j^-, m_j^+\right\}$. Now, the equivalent rainflow cycle s_j^{rfc} associated with M_j is given by

$$s_j^{\text{rfc}} = M_j - m_j^{\text{rfc}}. \tag{4.4}$$

The principle is depicted in figure 4.2, not so loosely inspired by [Gro00]. Notice that only the stress history maxima and minima are used in the rainflow cycle definition. A direct implementation of (4.4) could be outlined as follows:

1. Convert the stress history into an extremum sequence of alternating maxima and minima.
2. For each local maximum M_j, search the extremum sequence to identify the left and right regions in which to search for m_j^- and m_j^+.
3. For each local maximum M_j, search for m_j^- and m_j^+ and let
$$m_j^{\text{rfc}} = \max\left\{m_j^-, m_j^+\right\}. \tag{4.5}$$
4. For each local maximum M_j, compute $s_j^{\text{rfc}} = M_j - m_j^{\text{rfc}}$.

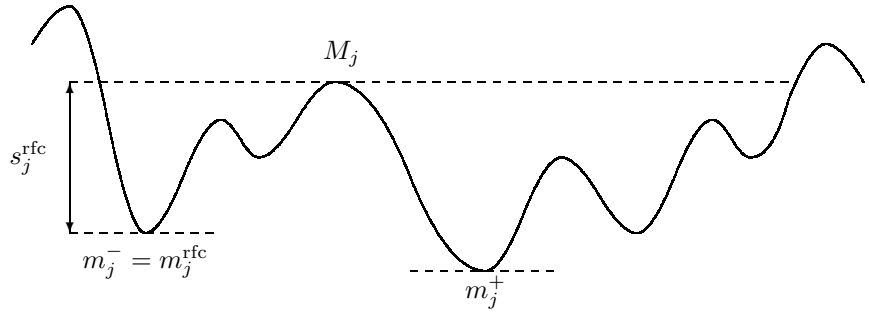

Figure 4.2: Definition of the equivalent rainflow cycle.

4.3.1 The four-point algorithm

The direct implementation of the rainflow counting algorithm described in the previous section clearly involves a lot of search activity. Another, and more efficient approach, results from a closer inspection of the equivalent rainflow cycles as defined by (4.4). First, we will define the term *inner range*:

Inner range Consider the quadruple $\langle x_1, x_2, x_3, x_4 \rangle$. The pair $\langle x_2, x_3 \rangle$ is said to form an inner range if the interval $[x_2; x_3]$ is contained in the interval $[x_1; x_4]$, i.e: $[x_2; x_3] \subseteq [x_1; x_4]$.

Consider the stress time series of figure 4.2, with all rainflow cycle pairs $\left\langle M_i, m_i^{\mathrm{rfc}} \right\rangle$ identified as depicted in figure 4.3. The rainflow count pairing can be thought of as an *extraction of inner ranges*. Specifically, consider the pairing $\left\langle M_{j-1}, m_{j-1}^{\mathrm{rfc}} \right\rangle$. This range pair can be thought of as an inner range in the outer range pair $\left\langle M_j, m_j^{\mathrm{rfc}} \right\rangle$, which, in turn, is an inner range of the range $\langle M_{j-2}, m_{j+2} \rangle$[1]. Similarly, the range pairs $\left\langle M_{j+1}, m_{j+1}^{\mathrm{rfc}} \right\rangle$ and $\left\langle M_{j+2}, m_{j+2}^{\mathrm{rfc}} \right\rangle$ are inner range pairs of the range $\langle M_{j+3}, m_{j+2} \rangle$. Finally, one should notice that removing an inner range pair from the time series does not change the status of the outer range embracing the inner range pair removed. As an example, removing the range pair $\left\langle M_{j-1}, m_{j-1}^{\mathrm{rfc}} \right\rangle$ would leave the range pair $\left\langle M_j, m_j^{\mathrm{rfc}} \right\rangle$ still being an inner range in $\langle M_{j-2}, m_{j+2} \rangle$.

The notion of inner ranges can be used for constructing an algorithm known as the *four-point algorithm* for rainflow cycle extraction. After converting the stress

[1]Notice that the minimum m_{j+2} does not have a superscripted "rfc" because it does not constitute a rainflow range pair with any of the maxima in the depicted stress history.

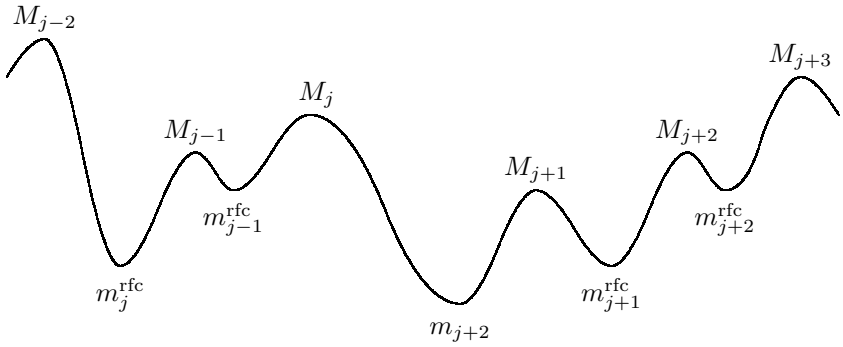

Figure 4.3: Rainflow count pairs in the stress history. The pairings can be interpreted as *inner range pairs*.

history to a series of extrema consisting of alternating maxima and minima, the algorithm proceeds as a two-stage process:

1. Extract all inner ranges, each representing af rainflow cycle range. This will leave a residual sequence with no inner ranges.
2. Handle residual. A new sequence is created by concatenating the residual with itself. Next, all inner ranges in the resulting sequence are extracted.

A description including detailed indexing and loop structures can be found in [Mad99]. Furthermore, an implementation of the four-point algorithm is found in appendix B.4.

We will illustrate the four-point algorithm with two examples—one example showing how a complex stress history is dissected into equivalent cycles, and one example demonstrating that the algorithm gives the correct result for the sinusoidal stress history on which the SN curve is based.

4.3.2 Example—complex stress history

Consider the stress history of figure 4.3 converted to a sequence of minima and maxima. This results in the sequence depicted in the upper left part of figure 4.4, denoted “1”.

In the figure, the first quadruple of extremes is marked. The algorithm checks

Step	Action	s^{rfc}
1	First extremum quadruple considered—no inner range.	
2	Quadruple +1 —no inner range.	
3	Quadruple +1 —inner range with magnitude 1 removed.	1
4	Quadruple -1 —inner range with magnitude 4 removed.	4
5	Quadruple -1 —no inner range.	
6	Quadruple +1 —no inner range.	
7	Quadruple +1 —inner range with magnitude 2 removed.	2
8	Quadruple -1 —no inner range.	
9	Quadruple +1 —inner range with magnitude 1 removed.	1
10	Residual sequence with no inner ranges	
11	Residual concatenated with itself.	
12	First extremum quadruple considered—no inner range.	
13	Quadruple +1 —no inner range.	
14	Quadruple +1 —inner range with magnitude 1 removed.	1
15	Quadruple -1 —inner range with magnitude 7 removed.	7
16	Residual sequence with no inner ranges	

Table 4.1: Steps in the complex stress history example. Upper part: inner range extraction (see figure 4.4). Lower part: residual handling (see figure 4.5)

if the two middle extremes forms an inner range. This is not the case, and the algorithm moves the quadruple one step forward (step 2), and looks for an inner range. This is not the case for the second quadruple either, so the quadruple is moved one step forward, giving the situation depicted in step 3.

Here, an inner range of unity width—marked with red crosses—is found, and the pair is removed from the sequence, yielding the sequence shown in step 4. Note that the quadruple starting point is now moved one step *backwards*. This is done to ensure that the new inner range that emerged upon removal of the present inner range pair, is detected by the algorithm. In step 4, the marked inner range of width 4 is removed.

The inner range extraction now proceeds as shown in the rest of figure 4.4 until only a residual sequence containing no inner ranges is left, cf. step 10. The procedure is summarised in the upper part of table 4.1.

The next stage of the four-point algorithm is handling of the residual, depicted in figure 4.5. First, the residual sequence is concatenated with itself, yielding the sequence shown in step 11. Careful inspection of the junction of the two sequences reveals that the required pattern of the slope changing sign at each entry in the sequence is broken. Therefore, the intermediate point in the junction

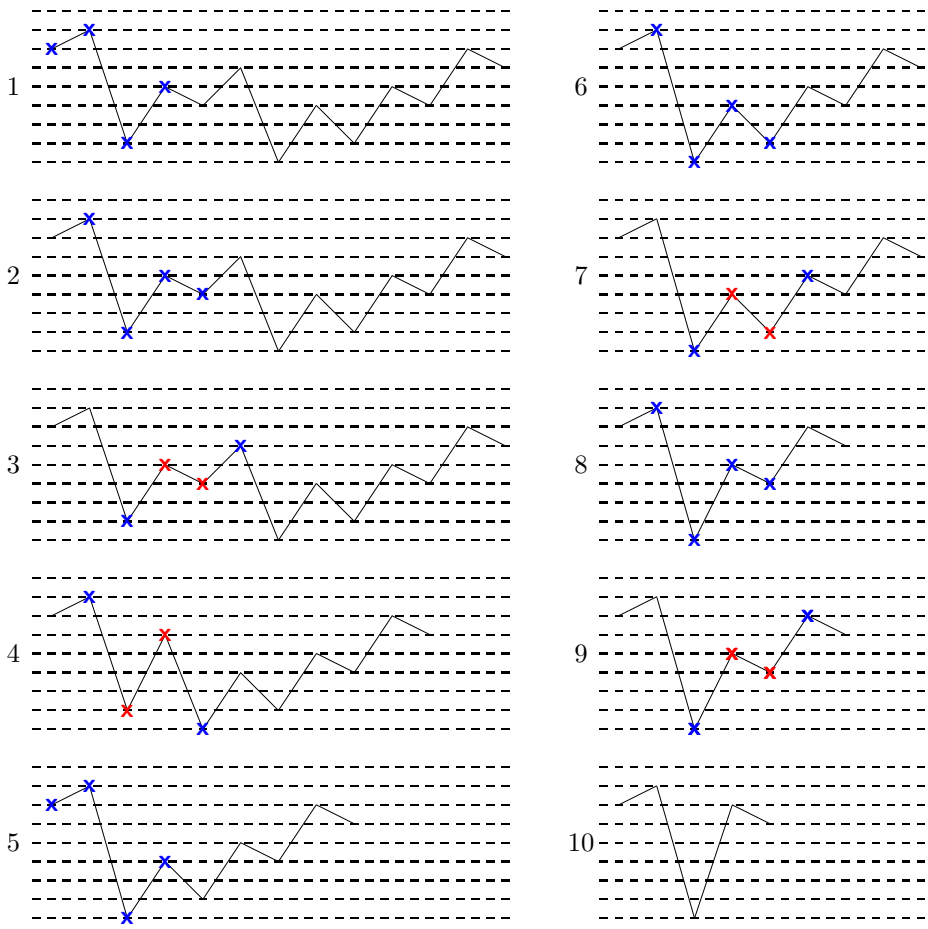

Figure 4.4: The four point algorithm, stage 1—extraction of inner ranges.

is removed, giving the sequence shown in step 12.

Next, an inner range removal procedure is carried out on the remaining sequence until a new residual arise (step 16), which concludes the four-point algorithm. The handling of the residual is summarised in the lower part of table 4.1.

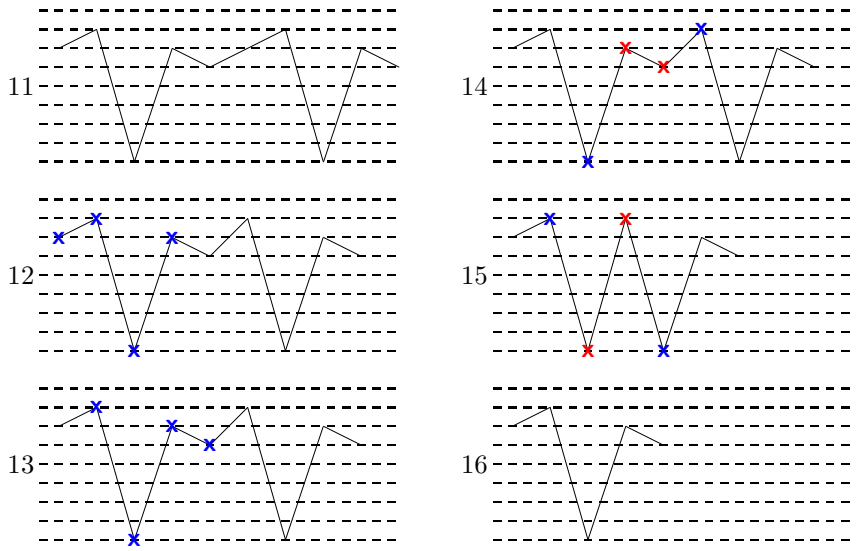

Figure 4.5: The four point algorithm, stage 2—handling the residual.

Combining the results of table 4.1 yields the total, equivalent set $\mathcal{S}$ of rainflow cycle ranges for the stress history used in the example:

$$\mathcal{S} = \{1, 4, 2, 1, 1, 7\}.$$

That is, a stress history consisting of three ranges with magnitude 1, one range with magnitude 2, one range with magnitude 4, and one range with magnitude 7 should cause the same amount of fatigue damage as the original stress history. The total damage would follow from Palmgren-Miner's damage rule as follows:

$$D = \frac{1}{K}\left(1^k + 4^k + 2^k + 1^k + 1^k + 7^k\right).$$

It should be mentioned that the handling of the residual can be done in several ways. The technique used here is denoted the Cloormann/Seeger method, which is also the default method used in the WAFO toolbox.[Gro00]

4.3.3 Example—sinusoidal stress history

To verify the connection between inner range pairs and equivalent cycles, consider a material with the SN curve depicted in figure 4.1 ($k = 4$, $K = 10^{32}$) being exposed to the sinusoidal stress history depicted in figure 4.6(a), consisting of 10^4 full, sinusoidal cycles. With the crosses depicting the first inner range encountered in the sequence, the figure illustrates that $10^4 - 1 = 9999$ inner ranges if magnitude 10^7 Pa can be extracted, leaving the residual seen in figure 4.6(b). Further, figure 4.6(c) illustrates that the concatenated residual sequence yields 1 inner cycle, giving a total of $9999 + 1 = 10^4$ equivalent cycles, each with magnitude 10^7. Applying Palmgren-Miner's damage rule gives

$$D = \sum_{i=1}^{M} \frac{1}{K} \left(s_i^{\mathrm{rfc}}\right)^k = \sum_{i=1}^{10^4} \frac{1}{10^{32}} \left(10^7\right)^4 = 1.$$

That is, unity damage as expected from the definition of the SN curve.

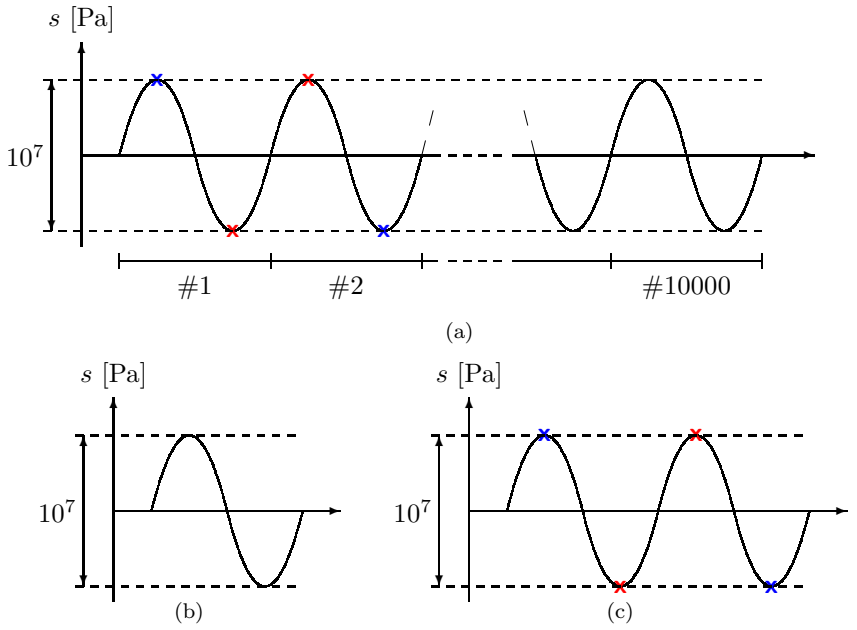

Figure 4.6: (a) $10^4 - 1 = 9999$ inner ranges of width 10^7 can be extracted from the stress history. (b) The residual. (c) Inner range removal in the concatenated residual yields one inner range of width 10^7, giving a total equivalent range count of exactly 10^4.

4.4 Expected damage rate

In practical applications, the set of equivalent rainflow ranges $\mathcal{S}$ is often binned to form a histogram of stress ranges. Letting $H_i^{\mathcal{S}}$ denote the number of stress ranges that fall into the i'th bin in an M-bin histogram, the Palmgren-Miner sum giving the total fatigue damage can be written as

$$D \approx \frac{1}{K} \sum_{i=1}^{M} s_i^k H_i^{\mathcal{S}}, \tag{4.6}$$

where the approximation sign indicates that the result is based on *binned* values of the rainflow ranges (binning into a finite number of levels introduce a quantization error). Note, that in (4.6), s_i denotes the center value of the histogram bins.

Now, assume that the number of cycles N_c in the stress history and the number of bins in the histogram approaches infinity. In this case, the histogram will approach the probability density function $p_s(s)$ for the rainflow ranges multiplied by the number of cycles, and the summation can be replaced by an integration[2]:

$$\lim_{N_c \to \infty} D = \frac{1}{K} \int_0^\infty s^k N_c p_s(s) ds = \frac{N_c}{K} \int_0^\infty s^k p_s(s) ds.$$

Not surprisingly, this result states that the damage will approach infinity when the number of cycles approaches infinity. Therefore, we will define the expected *damage pr. cycle*, d_c, as follows:

$$E[d_c] = \lim_{N_c \to \infty} \frac{D}{N_c} = \frac{1}{K} \int_0^\infty s^k p_s(s) ds. \tag{4.7}$$

Similarly, we will define the expected *damage pr. time unit*, d:

$$E[d] = d_c \nu_c = \frac{\nu_c}{K} \int_0^\infty s^k p_s(s) ds, \tag{4.8}$$

where ν_c denotes the number of cycles per unit time. Note that the damage rates are proportional to the k'th moment of the stress range density function. The quantity d is interesting in the sense that it equals the inverse of the expected lifetime:

$$E[L] = \frac{1}{E[d]}. \tag{4.9}$$

Finally, we will state a useful corollary relating the damage rates of two proportional stress histories. Assume the stress histories $s_1(t)$ and $s_2(t)$:

$$s_2(t) = a s_1(t),$$

[2]The integration limits of 0 and ∞ follows from the stress ranges—and thus the domain of their probability density functions—always being positive.

then the damage rates d_1 and d_2 are related as

$$d_2 = a^k d_1. \tag{4.10}$$

The proof follows readily from (4.2) or, alternatively, from considering the density functions of $s_1(t)$ and $s_2(t)$ and evaluate (4.8).

Chapter 5

Estimating fatigue damage from spectral properties

Chapter 4 described a rather complicated, algorithmic relation between a given load history and the resulting fatigue damage. As of the time of writing, no theoretical results exist that provide a closed-form result for the fatigue damage. As a result, attention is drawn towards approximations giving a fatigue damage estimate in terms of the spectral properties of the stress history applied.

To provide a spectral description of a scalar, stochastic process $x(t)$, we will define the spectral density $S_X(\omega)$ as the Fourier transform of the autocovariance function $R_X(\tau)$, giving the Fourier transform pair

$$S_X(\omega) = \int_{-\infty}^{\infty} R_X(\tau) e^{-j\omega\tau} d\tau$$
$$R_X(\tau) = \frac{1}{2\pi} \int_{-\infty}^{\infty} S_X(\omega) e^{j\omega\tau} d\omega.$$

An important relation arise when considering the variance σ_X^2 (assuming a zero-mean $x(t)$):

$$\sigma_X^2 = R_X(0) = \frac{1}{2\pi} \int_{-\infty}^{\infty} S_X(\omega) d\omega.$$

Now, we will define the mth spectral moment λ_m as follows:

$$\lambda_m \equiv \frac{1}{\pi} \int_0^\infty \omega^m S_X(\omega) d\omega. \tag{5.1}$$

Three important properties follow from this definition:

$$\lambda_0 = \sigma_X^2 \tag{5.2a}$$
$$\lambda_2 = \sigma_{\dot{X}}^2 \tag{5.2b}$$
$$\lambda_4 = \sigma_{\ddot{X}}^2. \tag{5.2c}$$

That is, the variance is given by the 0th spectral moment, which readily follows from (5.1). Further, the variance of the first derivative of $x(t)$ is given by the second spectral moment, and finally, the variance of the second derivative of $x(t)$ is given by the fourth spectral moment. The derivations of the latter two properties are straightforward and can be seen in e.g. [New84].

The 1st spectral moment, λ_1, does not lend itself to a nice, intuitive interpretation as do the 0th, 2nd, and 4th spectral moments. One can show that the first spectral moment can be interpreted as the covariance between a signal and the Hilbert transform of it's first derivative, but this is of little practical use.

5.1 Damage in a narrow-band process

The size of the equivalent cycles extracted by the rainflow counting algorithm depend, as demonstrated in chapter 4, on the waveform of the stress history in a rather complicated manner. Rychlik has shown, though, that the expected damage rate for a stationary, Gaussian stress history with a given variance is bounded from above by the damage rate for a narrow-band process with the same variance:

$$E[d] \leq E[d_{\curlywedge}], \tag{5.3}$$

where $d_{\curlywedge}$ denotes the damage rate for the narrow-band process[1].

Now, we will investigate the damage rate of a narrow-band process. The results presented here follow from [Ryc93, BT05].

A *narrow-band approximation* of a Gaussian process is defined as a narrow-banded, Gaussian process with the same variance σ^2 as the original process.

[1] The symbol $\curlywedge$ indicates the peaky spectrum for a narrow-band process.

The peak amplitudes of such a process are Rayleigh distributed, thus having a probability density function $p_a(a)$ given by:

$$p_a(a) = \frac{a}{\sigma^2} e^{-\frac{a^2}{2\sigma^2}} = \frac{a}{\lambda_0} e^{-\frac{a^2}{2\lambda_0}} \quad , \quad a \geq 0.$$

Now, consider the example narrow-band process depicted in figure 5.1. When performing a rainflow counting procedure in such a process, all peaks will pair to a through with similar amplitude, producing inner ranges with magnitudes twice the amplitudes of the narrow-band process. As a result, the rainflow-counting procedure will result in an equivalent range density $p_s(s)$ related to $p_a(a)$ as:

$$p_s(s) = \frac{1}{2} p_a\left(\frac{s}{2}\right) = \frac{1}{2} \frac{s/2}{\lambda_0} e^{-\frac{(s/2)^2}{2\lambda_0}} = \frac{s}{4\lambda_0} e^{-\frac{s^2}{8\lambda_0}} \quad , \quad s \geq 0. \tag{5.4}$$

That is, when the range s is considered instead of the amplitude, the probability density function is stretched (the $\frac{s}{2}$ thing) and it's magnitude is halfed to ensure that the density integrates to unity.

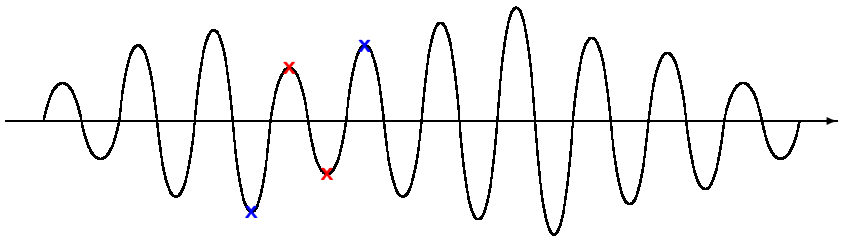

Figure 5.1: A narrow-band process. When extracting equivalent cycles, the distribution for the inner ranges will, except for a scaling factor of 2, equal the distribution of the peak amplitudes.

For a Gaussian process, the number of peaks per unit time, denoted ν_p, is given by:

$$\nu_p = \frac{1}{2\pi} \sqrt{\frac{\lambda_4}{\lambda_2}}. \tag{5.5}$$

As the rainflow counting procedure will create one equivalent cycle for each peak, the number of cycles per unit time equals the number of peaks per unit time:

$$\nu_c = \nu_p$$

Now, inserting (5.4) and (5.5) into (4.8) yields the expected damage rate d_λ for a narrowband process:

$$d_\lambda = \frac{1}{2\pi} \sqrt{\frac{\lambda_4}{\lambda_2}} \frac{1}{K} \int_0^\infty s^k \frac{s}{4\lambda_0} e^{-\frac{s^2}{8\lambda_0}} ds \tag{5.6}$$

In (5.6), the integral is recognised as the k'th moment of a Rayleigh density function with parameter $\sqrt{4\lambda_0}$. In general, the k'th moment μ_k of a Rayleigh density with parameter x is given by:

$$\mu_k = x^k 2^{k/2} \Gamma\left(1 + \frac{k}{2}\right), \tag{5.7}$$

where $\Gamma(\cdot)$ denotes the Gamma function. Inserting this result in (5.6) finally yields:

$$\begin{aligned} d_{\lightning} &= \frac{1}{2\pi}\sqrt{\frac{\lambda_4}{\lambda_2}}\frac{1}{K}\sqrt{4\lambda_0}^k 2^{k/2}\Gamma\left(1+\frac{k}{2}\right) \\ &= \frac{1}{2\pi}\sqrt{\frac{\lambda_4}{\lambda_2}}\frac{1}{K}\left(2\sqrt{2\lambda_0}\right)^k \Gamma\left(1+\frac{k}{2}\right). \end{aligned} \tag{5.8}$$

Finally, it should be emphasized that this result is only valid for narrow-banded, *Gaussian* stress histories. As an example, it can easily be shown that attempting to compute the damage rate of a sinusoidal stress history (which is narrow-banded, but *not* Gaussian) using (5.8) will result in an estimate that is off by a factor of $\Gamma\left(1+\frac{k}{2}\right)$.

5.2 Benasciutti's approximation

As mentioned earlier, the narrowband solution presented in the previous section provides a conservative estimate for the damage rate for a process that is not a narrow-band process, cf. (5.3).

In [BT05], Benasciutti proposes an estimate of the expected fatigue damage rate given as the narrow-band approximation modified by a correction factor to account for the process not necessarily being narrow-band:

$$E[d] \approx \underbrace{\frac{1}{2\pi}\sqrt{\frac{\lambda_4}{\lambda_2}}\frac{1}{K}\left(2\sqrt{2\lambda_0}\right)^k \Gamma\left(1+\frac{k}{2}\right)}_{d_{\lightning}} \underbrace{\left(b + (1-b)\alpha_2^{k+1}\right)}_{\text{Correction}} \tag{5.9a}$$

where

$$b = \frac{(\alpha_1 - \alpha_2)\left[1.112\left(1 + \alpha_1\alpha_2 - (\alpha_1+\alpha_2)\right)e^{2.11\alpha_2} + (\alpha_1 - \alpha_2)\right]}{(\alpha_2 - 1)^2} \tag{5.9b}$$

and

$$\alpha_1 = \frac{\lambda_1}{\sqrt{\lambda_0\lambda_2}} \quad , \quad \alpha_2 = \frac{\lambda_2}{\sqrt{\lambda_0\lambda_4}}. \tag{5.9c}$$

Again, as the Benasciutti approximation makes use of the narrowband approximation, it should be emphasized that the approximation is valid for Gaussian stress histories only.

For notational convenience we will use the symbol Λ_x to denote the Benasciutti damage rate estimate for the process $x(t)$.[2]

To summarise, equation (5.9) provides a means of estimating the fatigue damage rate using only the material constants K and k, and the spectral moments λ_0, λ_1, λ_2, and λ_4 for the stress history. That is, no time series for the stress history is needed.

As will be shown in chapter 6, the spectral moments for the stress histories in a linear system model can be computed very efficiently, thus providing a means of predicting fatigue damage in linear models.

[2]The greek letter Λ here indicates an altered, or corrected, version of the narrowband symbol λ

Chapter 6

Damage estimation in the wind turbine model

The results presented in chapter 5 show that fatigue damage can be estimated from four spectral moments of the stress history. Thus, if we can compute the spectral moments of the stress histories in the wind turbine components, we can estimate their expected lifetimes.

First, in section 6.1, an important result will be presented, providing efficient computation of the spectral moments in linear models.

The wind turbine model considered in this project comprises three fatigue inducing mechanisms: the pitch system, the low-speed shaft, and the flexible tower structure. Computing the stress histories for these components is the subject of sections 6.2 and 6.3.

6.1 Computing spectral moments in linear models

Consider a stochastic process $x(t)$ resulting from of a white noise process $\varepsilon(t)$ with intensity σ_ε^2 being filtered through a rational, stable transfer function $H(s)$.

The spectral density $S_X(\omega)$ of the process $x(t)$ can be described in terms of the transfer function as follows:

$$S_X(\omega) = H(j\omega)H(-j\omega)\sigma_\varepsilon^2. \tag{6.1}$$

As $H(s)$ is a rational function in s with real coefficients, it can be written on the pole-zero-gain form

$$H(s) = k\frac{\prod_{\mathcal{Z}_\mathbb{R}}(s - z_i)\prod_{\mathcal{Z}_\mathbb{C}}\left(s^2 - 2\Re(z_i)\, s + |z_i|^2\right)}{\prod_{\mathcal{P}_\mathbb{R}}(s - p_i)\prod_{\mathcal{P}_\mathbb{C}}\left(s^2 - 2\Re(p_i)\, s + |p_i|^2\right)},$$

where $\mathcal{Z}_\mathbb{R}$ denotes the set of real zeros in $H(s)$, and $\mathcal{Z}_\mathbb{C}$ denotes the set of complex conjugated zero pairs. Similarly, $\mathcal{P}_\mathbb{R}$ denotes the set of real poles in $H(s)$, and $\mathcal{P}_\mathbb{C}$ denotes the set of complex conjugated pole pairs.

Now, consider $H(j\omega)$ and $H(-j\omega)$:

$$\begin{aligned}
H(j\omega) &= k\frac{\prod_{\mathcal{Z}_\mathbb{R}}(j\omega - z_i)\prod_{\mathcal{Z}_\mathbb{C}}\left(-\omega^2 - j2\Re(z_i)\,\omega + |z_i|^2\right)}{\prod_{\mathcal{P}_\mathbb{R}}(j\omega - p_i)\prod_{\mathcal{P}_\mathbb{C}}\left(-\omega^2 - j2\Re(p_i)\,\omega + |p_i|^2\right)} \\
H(-j\omega) &= k\frac{\prod_{\mathcal{Z}_\mathbb{R}}(-j\omega - z_i)\prod_{\mathcal{Z}_\mathbb{C}}\left(-\omega^2 + j2\Re(z_i)\,\omega + |z_i|^2\right)}{\prod_{\mathcal{P}_\mathbb{R}}(-j\omega - p_i)\prod_{\mathcal{P}_\mathbb{C}}\left(-\omega^2 + j2\Re(p_i)\,\omega + |p_i|^2\right)}.
\end{aligned}$$

As the following identities holds true:

$$(j\omega - z)(-j\omega - z) = \omega^2 + z^2$$

and

$$\left(-\omega^2 - j2\Re(z)\,\omega + |z|^2\right)\left(-\omega^2 + j2\Re(z)\,\omega + |z|^2\right) = \omega^4 + 2\left(\Re(z)^2 - \Im(z)^2\right)\omega^2 + |z|^4,$$

equation (6.1) can be written as

$$S_X(\omega) = k^2\frac{\prod_{\mathcal{Z}_\mathbb{R}}(\omega^2 + z_i^2)\prod_{\mathcal{Z}_\mathbb{C}}\left(\omega^4 + 2\left(\Re(z_i)^2 - \Im(z_i)^2\right)\omega^2 + |z_i|^4\right)}{\prod_{\mathcal{P}_\mathbb{R}}(\omega^2 + p_i^2)\prod_{\mathcal{P}_\mathbb{C}}\left(\omega^4 + 2\left(\Re(p_i)^2 - \Im(p_i)^2\right)\omega^2 + |p_i|^4\right)} \equiv k^2\frac{P(\omega)}{Q(\omega)}\sigma_\varepsilon^2. \tag{6.2}$$

It is easily verified that the set of poles $\mathcal{P}$ in $H(s)$ are related to the poles $\hat{\mathcal{P}}$ in $S_X(\omega)$ as follows:

$$\hat{\mathcal{P}} = j\mathcal{P} \cup j\mathcal{P}^*.$$

This mapping is depicted in 6.1 and has two important properties:

- A real pole in $H(s)$ maps to a pair of purely imaginary, complex conjugated poles in $S_X(\omega)$.
- A complex conjugated pole pair in $H(s)$ maps to a quadruple of poles in $S_X(\omega)$ distributed symmetrically around both the real axis and the imaginary axis.

Notice that this mapping corresponds to rotating the poles of $H(s)$ 90° around the origin in the complex plane and mirror them in the real axis. Further, notice that $S_X(\omega)$ will have no real poles. Also, notice that $P(\omega)$ and $Q(\omega)$ will have even-ordered terms only.

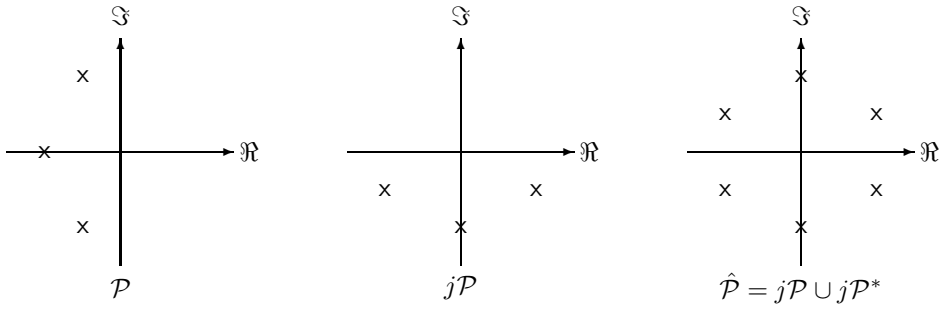

Figure 6.1: The set of poles $\mathcal{P}$ of the transfer function $H(s)$ maps into the set of poles $\hat{\mathcal{P}}$ in the rational function $S_X(\omega)$ defining the spectral density. Real poles map into pairs located on the imaginary axis and complex conjugated pole pairs map into quadruples of poles symmetrical in both the real and the imaginary axis.

Now, we turn our attention towards the evaluation of the integral defining the spectral moments, cf. equation (5.1). First, we present an important result from calculus [SM92]:

$$\int \frac{B(x)}{A(x)}dx = \sum_{\mathcal{A}_{\mathbb{R}}} \frac{B(a)}{A'(a)} \log|x-a| + \sum_{\mathcal{A}_{\mathbb{C}}} \left\{ \Re\left(\frac{B(a)}{A'(a)}\right) \log\left|(x-\Re(a))^2 + (\Im(a))^2\right| - 2\Im\left(\frac{B(a)}{A'(a)}\right) \arctan\left(\frac{x-\Re(a)}{\Im(a)}\right) \right\}, \quad (6.3)$$

where $\mathcal{A}_{\mathbb{R}}$ denotes the set of real roots a in $A(x)$ and $\mathcal{A}_{\mathbb{C}}$ denotes the set of complex conjugated root pairs $\langle a, a^* \rangle$ in $A(x)$. $A'(x)$ denotes the derivative of $A(x)$ with respect to x.

Now, consider the m'th spectral moment λ_m:

$$\lambda_m = \frac{1}{\pi}\int_0^\infty \omega^m S(\omega)d\omega = \frac{k^2\sigma_\varepsilon^2}{\pi}\int_0^\infty \frac{\omega^m P(\omega)}{Q(\omega)}d\omega = \frac{k^2\sigma_\varepsilon^2}{\pi}\int_0^\infty \frac{\bar{P}(\omega)}{Q(\omega)}d\omega. \quad (6.4)$$

In order to evaluate this integral using (6.3), we note some useful properties. First, recalling that $Q(\omega)$ has complex conjugated roots only, the first sum of (6.3) will not contribute to the integral (6.4). Thus:

$$\int \frac{\bar{P}(\omega)}{Q(\omega)}d\omega = \sum_{p\in j\mathcal{P}}\left\{\Re\left(\frac{\bar{P}(p)}{Q'(p)}\right)\log\left|(\omega-\Re(p))^2+(\Im(p))^2\right| \right.$$
$$\left. -2\Im\left(\frac{\bar{P}(p)}{Q'(p)}\right)\arctan\left(\frac{\omega-\Re(p)}{\Im(p)}\right)\right\}, \quad (6.5)$$

where, as indicated, the sum runs over all *pairs* of poles in $S_X(\omega)$. Next, consider (6.5) evaluated in the limit $\omega\to\infty$:

$$\lim_{\omega\to\infty}\sum_{p\in j\mathcal{P}}\left\{\Re\left(\frac{\bar{P}(p)}{Q'(p)}\right)\log\left|(\omega-\Re(p))^2+(\Im(p))^2\right| \right.$$
$$\left. -2\Im\left(\frac{\bar{P}(p)}{Q'(p)}\right)\arctan\left(\frac{\omega-\Re(p)}{\Im(p)}\right)\right\}$$
$$= 2\log|\omega|\sum_{p\in j\mathcal{P}}\Re\left(\frac{\bar{P}(p)}{Q'(p)}\right) - 2\sum_{p\in j\mathcal{P}}\Im\left(\frac{\bar{P}(p)}{Q'(p)}\right)\frac{\pi}{2}.$$

We will, without proof, claim that

$$\sum_{p\in j\mathcal{P}}\Re\left(\frac{\bar{P}(p)}{Q'(p)}\right) = 0, \quad (6.6)$$

which yields, for the upper integration limit:

$$\int \frac{\bar{P}(\omega)}{Q(\omega)}d\omega\Big|_{\omega=\infty} = -\pi\sum_{p\in j\mathcal{P}}\Im\left(\frac{\bar{P}(p)}{Q'(p)}\right).$$

For the lower integration limit $\omega = 0$ we readily have:

$$\int \frac{\bar{P}(\omega)}{Q(\omega)}d\omega\Big|_{\omega=0} = \sum_{p\in j\mathcal{P}}\left\{\Re\left(\frac{\bar{P}(p)}{Q'(p)}\right)\log|p|^2 + 2\Im\left(\frac{\bar{P}(p)}{Q'(p)}\right)\arctan\left(\frac{\Re(p)}{\Im(p)}\right)\right\},$$

which gives, for the definite integral:

$$\int_0^\infty \frac{\bar{P}(\omega)}{Q(\omega)}d\omega = -\pi\sum_{p\in j\mathcal{P}}\Im\left(\frac{\bar{P}(p)}{Q'(p)}\right)$$
$$-\sum_{p\in j\mathcal{P}}\left\{\Re\left(\frac{\bar{P}(p)}{Q'(p)}\right)\log|p|^2 + 2\Im\left(\frac{\bar{P}(p)}{Q'(p)}\right)\arctan\left(\frac{\Re(p)}{\Im(p)}\right)\right\}. \quad (6.7)$$

This result allows us to state the following algorithm for computing the mth spectral moment of the process $x(t)$:

Given the poles p, zeros z, and the gain k of the generating transfer function $H(s)$ as well as the variance σ_ε^2 of the driving noise,

1. Form polynomium $P(\omega)$. Exploiting symmetry, this can be done efficiently by first sorting the zeros into real zeros and complex conjugated zero pairs. $P(\omega)$ is now formed by successively multiplying polynomiums of the form $\omega^2 - z_i^2$ (real zeros) and $\omega^4 + 2\left(\Re(z_i)^2 - \Im(z_i)^2\right)\omega^2 + |z_i|^4$ (complex conjugated zero pairs), cf. (6.2). In MATLAB, this is done by convolving coefficient vectors.

2. Similarly, form polynomium $Q(\omega)$ from the poles, exploiting symmetry in the same way as for $P(\omega)$.

3. Compute $Q'(\omega)$. For this purpose, MATLAB provides the function `polyder`.

4. Form $\bar{P}(\omega) = \omega^m P(\omega)$. In MATLAB, this is done by appending m zeros to the coefficient vector for $P(\omega)$.

5. Compute the poles a over which the summation in (6.7) should run: $a_i = jp_i$.

6. Compute
$$\lambda_m = \frac{k^2\sigma_\varepsilon^2}{\pi}\int_0^\infty \frac{\bar{P}(\omega)}{Q(\omega)}d\omega,$$
using (6.7) for evaluation of the integral.

A MATLAB implementation of the spectral moment computation is shown in appendix B.2. Note that, for a given spectral density, the $Q(\omega)$ polynomium used for computing λ_m is independent of m. Therefore, the contribution $Q'(p)$ can be evaluated before entering the loop where the integrals defining the desired spectral moments are evaluated.

Further, note that the algorithm checks if the integral exist. The relative order of $\bar{P}(\omega)/Q(\omega)$ must be at least two for the integral to converge. This implies that the m'th spectral moment is only defined for systems with a relative order higher than or equal to $\left\lceil \frac{m}{2} + 1 \right\rceil$, where $\lceil x \rceil$ denotes the smallest integer larger than or equal to x.

6.2 Tower and drive train fatigue

We will assume that the components under consideration (the main shaft and the tower) are not exposed to stresses that exceed the limit under which a linear relationship exists between stress and strain (deformation). That is:

$$\begin{aligned} s_z &= C_z z \\ s_\theta &= C_\theta \theta \end{aligned} \tag{6.8a}$$

where the constants C_z and C_θ are functions of the specific geometries involved. This assumption results in the following relations:

$$\begin{aligned} \lambda_m^{s_z} &= C_z^2 \lambda_m^z \\ \lambda_m^{s_\theta} &= C_\theta^2 \lambda_m^\theta \end{aligned} \tag{6.8b}$$

and

$$\begin{aligned} \Lambda_{s_z} &= C_z^k \Lambda_z \\ \Lambda_{s_\theta} &= C_\theta^k \Lambda_\theta. \end{aligned} \tag{6.8c}$$

Relations (6.8b) result from the definition of spectral moments. Relations (6.8c) results from (6.8b) and (5.9), and allows the following approach to the fatigue damage estimation in the drive train and in the tower:

1. For the linear, stochastic wind turbine model (2.14), compute the transfer functions from the driving noise ε to the deformation state variables θ and z.
2. Use the spectral moment algorithm to compute the spectral moments (λ^z or λ^θ).
3. Compute Benasciutti's approximation using the computed spectral moments and the material properties k and K.
4. Multiply by C_z^k (tower) or C_θ^k (shaft) to obtain the damage rate estimate.

Where nothing else stated, we will use $k = 4$ and $K = 6.25 \cdot 10^{37}$ for the tower and the drive train.

6.3 Blade bearing fatigue

When the blades are pitched, the balls inside the blade bearings roll along the bearing rings. Thus, every single point in the bearings rings will experience a

quasi-cyclic stress history due to the balls rolling past them. The magnitude of the stress will be proportional to the blade root moment M_B. The frequency of the quasi-cyclic stress history will be proportional to the absolute value of the pitch angle rate of change, $|\dot{\beta}|$. These considerations imply that the fatigue damage D_β on the pitch bearings during a time interval T can be expressed in the form

$$D_\beta \propto \int_t^{t+T} M_B^k(t)|\dot{\beta}(t)|dt, \tag{6.9}$$

where k is the Wöhler exponential for the bearing material. For ball bearings the Wöhler exponential is typically $k = 3$. [BSJB01]

Now, in practice the variability of the blade root moment M_B is relatively low. That is, the standard deviation is small compared to the mean value. Therefore, we will make the rough approximation that the blade root moment is constant for a constant mean wind. Thus, (6.9) becomes:

$$D_\beta \propto \int_t^{t+T} |\dot{\beta}(t)|dt.$$

Defining the average pitch bearing damage rate as

$$d_\beta \equiv \lim_{T\to\infty} \frac{D_\beta}{T}$$

we get (assuming ergodicity):

$$d_\beta \propto \lim_{T\to\infty} \frac{1}{T} \int_t^{t+T} |\dot{\beta}(t)|dt = E\left[|\dot{\beta}(t)|\right]$$

Further, we will assume that the pitch angle rate of change is normally distributed with zero mean and variance $\sigma_{\dot{\beta}}^2$, thus having the probability density function

$$p_{\dot{\beta}}(\dot{\beta}) = \frac{1}{\sigma_{\dot{\beta}}\sqrt{2\pi}} \exp\left(-\frac{\dot{\beta}^2}{2\sigma_{\dot{\beta}}^2}\right) \quad , \quad -\infty < \dot{\beta} < \infty.$$

In turn, the probability density function $p_{|\dot{\beta}|}(\dot{\beta})$ for the absolute value of the pitch rate is given by

$$p_{|\dot{\beta}|}(\dot{\beta}) = \frac{2}{\sigma_{\dot{\beta}}\sqrt{2\pi}} \exp\left(-\frac{\dot{\beta}^2}{2\sigma_{\dot{\beta}}^2}\right) \quad , \quad 0 < \dot{\beta} < \infty.$$

Finally, the expectation value $E\left[|\dot{\beta}(t)|\right]$ is given by the first moment of $p_{|\dot{\beta}|}(\dot{\beta})$:

$$E\left[|\dot{\beta}(t)|\right] = \int_0^\infty \dot{\beta} \frac{2}{\sigma_{\dot{\beta}}\sqrt{2\pi}} \exp\left(-\frac{\dot{\beta}^2}{2\sigma_{\dot{\beta}}^2}\right) d\dot{\beta} = \sqrt{\frac{2}{\pi}}\sqrt{\sigma_{\dot{\beta}}^2}.$$

Recalling that the variance of the first derivative equals the 2nd spectral moment, we finally write:

$$d_\beta \propto E\left[|\dot{\beta}(t)|\right] \propto \sqrt{\lambda_2^\beta}$$

That is, the pitch bearing fatigue damage rate is expected to be proportional to the square root of the 2nd spectral moment for the pitch angle β. We will denote the proportionality constant C_β:

$$E\left[d_\beta\right] = C_\beta \sqrt{\lambda_2^\beta}. \tag{6.10}$$

Chapter 7

Controller design as a fatigue-estimate based optimisation problem

The results in the preceding chapters provide a means for efficient estimation of damage rates for load histories being linear in the state variables in a linear system. This, in turn, allows for efficient evaluation of a performance cost function involving fatigue damage. In this chapter we will demonstrate how this allows for numerical optimisation of a state-feedback controller for the wind turbine.

The treatment is restricted to a controller for a turbine model linearised around a mean wind $\bar{v} = 16$ m/s. That is, a mode IV controller like the one treated in section 3.4.

7.1 Defining the optimisation problem

We will define the controller design objective as follows:

Determine the controller U^* that maximizes the shortest component life time under the constraint that the variabilities of generator speed and produced power do not exceed the limits Γ_{ω_g} and Γ_{P_e}.

This can be formulated as a minimax optimisation problem as follows:

$$U^* = \underset{U}{\operatorname{argmin}} \left\{ \max_i d_i(U) \right\} \tag{7.1a}$$

subject to

$$c_1 : \quad c_{v,\omega_g}(U) - \Gamma_{\omega_g} < 0 \tag{7.1b}$$
$$c_2 : \quad c_{v,P_{\text{ref}}}(U) - \Gamma_{P_e} < 0, \tag{7.1c}$$

where d_i denotes the damage rate for the ith component. In the present case, three different damage rates are computed, namely the low-speed shaft damage rate d_θ, the tower damage rate d_z, and the pitch bearing damage rate d_β. Thus, the damage rate vector d is defined as

$$d \equiv \begin{pmatrix} d_\theta \\ d_z \\ d_\beta \end{pmatrix}. \tag{7.2}$$

The optimisation problem (7.1) is a very general formulation as the domain of the decision variable U spans all possible controllers—linear or nonlinear—with structure as well as parameters being decision variables. This implies an infinite-dimensional optimisation problem and is of no practical use.

Therefore, we will narrow down the domain of the decision variable to span the range of linear state-feedback controllers having the control law

$$\tilde{u} = -L\tilde{x}_a. \tag{7.3}$$

where the state vector $\tilde{x}_a$ includes the integral states, as defined in section 3.4. Thus, the dimension of the optimisation problem is determined by the number of entries in the gain matrix. In the present case, $L \in \mathbb{R}^{2\times 11}$, which implies a decision vector of dimension 22.

7.2 Solving the optimisation problem

The MATLAB optimization toolbox provides the function `fminimax` for solving nonlinear, constrained minimax problems. We will use this routine, even though it actually only finds local minima in the cost function. (The problem (7.1) is a global optimisation problem). For further elaboration on this subject, see section 7.7.2.

7.3 Evaluating the cost function

The vector-valued cost function returning the damage rate vector d is evaluated as follows:

Given the gain matrix L,

1. Form closed-loop description of the form $\dot{x} = A_{\text{cl}}x + w$.
2. Compute transfer functions from the driving noise process w to state variables θ, z, and β. The transfer functions should be expressed on pole-zero-gain form. In MATLAB , we suggest that the poles are found as `p=eig(Acl)` and that the gains and the zeros are found using `[k,z]=tzero(...)`.
3. Compute spectral moments for θ, z, and β using the algorithm stated in section 6.1.
4. Apply (5.9) and (6.8c) to estimate damage rates d_θ and d_z for shaft and tower and apply (6.10) to compute damage rate d_β for the pitch bearings.
5. Form output vector $d = \begin{pmatrix} d_\theta & d_z & d_\beta \end{pmatrix}^T$.

7.4 Evaluating the constraint functions

The constraint functions are based on the expected variabilities of the output signals ω_g and P_e, and are computed as follows:

Given the gain matrix L,

1. Compute state vector variance matrix P_x by solving the Lyapunov equation
$$A_{\text{cl}}P_x + P_x A_{\text{cl}}^T + R_1 = 0.$$
In MATLAB this is done using `Px=lyap(Acl,R1)`.
2. Compute output variance matrix $P_y = C_a P_x C_a^T$ and extract $\sigma^2_{\omega_g}$ and $\sigma^2_{P_e}$ from the diagonal.
3. Compute variabilities:
$$c_{v,\omega_g} = \frac{\sigma_{\omega_g}}{\bar{\omega}_g} \quad , \quad c_{v,P_e} = \frac{\sigma_{P_e}}{\bar{P}_e}.$$

4. Form constraint function values:

$$c_1 = c_{v,\omega_g} - \Gamma_{\omega_g} \quad , \quad c_2 = c_{v,P_e} - \Gamma_{P_e}.$$

7.4.1 An extra constraint

The integral states in the setup has the function of ensuring correct reference following despite model nonlinearities and uncertainties. This performance measure is *not* a part of the cost (7.1a). Solving (7.1) without further action results in the gains from the integral states being very small, leaving the closed-loop system with almost pure integrators. In a real-life, nonlinear setup this would effectively cancel the intended effect of the integral states.

To overcome this problem, an extra constraint is added, imposing an upper limit on the time constants in the closed-loop system, formulated as a maximum value for the largest real part of the closed-loop eigenvalues λ^{cl}:

$$c_3 : \quad \max_i \left\{ \Re \left(\lambda_i^{\text{cl}} \right) \right\} + \Gamma_\lambda < 0.$$

In the implementation, $\Gamma_\lambda = 0.01$, corresponding to an upper limit for the time constants of 100 seconds.

7.5 Normalised damage rates

As the absolute values of the proportionality factors C_z, C_θ, and C_β are not known (cf. sections 6.2 and 6.3), we will demonstrate the controller optimisation procedure using what could be denoted *normalised damage rates*.

The idea is as follows:

1. Form a closed-loop system description using the baseline LQI-controller described in section 3.4.
2. Assume proportionality constants $C_z = C_\theta = C_\beta = 1$, and compute expected damage rates for the three components. (Details are given in section 7.3).
3. Now, compute new values for C_z, C_θ, and C_β such that the three damage rates would equal $0.01 \frac{1}{365 \times 24 \times 60 \times 60}$. This is equivalent to applying the assumption that, using the baseline LQI-controller, the three components would have a lifetime of 100 years.

7.6 Design examples

We will demonstrate the controller optimisation by three examples:

Design example #1 Find the controller that minimizes the largest damage rate subject to the variabilities of the generator speed and the produced power not exceeding 0.01.

Design example #2 Assume that the requirement for constant power is relaxed such that the variability of the power is allowed to reach 0.05, and find the optimal controller.

Design example #3 Assume that the tower base diameter D is increased by 10%. As the stress s_t at the tower base is related to the tower deflection z approximately as $s_t \propto \frac{z}{D^4}$, the increase in diameter would decrease C_z by a factor of $1.1^4 = 1.46$. Find the optimal controller using the new C_z value.

The results of these design examples are summarised in table 7.1. The estimated values in the second column are the life times in years resulting from evaluating the cost function at the solution, and the expected variabilities resulting from evaluation of the constraint functions at the solution. That is, the life times equal the Benasciutti estimates and the variabilities are analytic results, as described in sections 7.3 and 7.4.

The simulation values in the third column result from simulating 10,000 seconds of operation of the closed-loop system using the gain resulting from the minimax optimisation.

The simulated damage rates for the tower and for the drive train are computed from the simulation output as described in chapter 4. That is, using the four-point algorithm and Palmgren-Miner's damage rule. The stress histories s_z and s_θ are obtained by multiplying the z and θ trajectories with C_z and C_θ, respectively, cf. (6.8a). The simulated damage rate for the pitch bearings is found by computing the sample standard deviation of $\dot{\beta}$ and multiply by C_β, cf. (6.10).

The simulated variabilities are computed directly from the sample standard deviations and mean values of the ω_g and P_e trajectories.

Finally, the last column states the percentage by which the expected values differ from the simulation result. This has special interest when it comes to the estimated life times as these result from the Benasciutti approximation.

For illustrative purposes, the first 50 seconds of each simulation are depicted in figure 7.1.

A few comments should be added to each of the design results.

Design example #1 The resulting variabilities for the generator speed and the produced power for the baseline controller indicates that there is room for improvement as these variabilities are only approximately one fourth of their upper limits 0.01. As the optimisation result shows, this overhead is turned into doubled lifetimes for all three components. Inspection of the simulation output in figure 7.1 shows that the higher lifetimes result from reduced pitch activity (smaller pitch rate), tower damping, and a lowering of the drive train deformations. The price paid is, as expected, larger fluctuations in generator speed and produced power. Finally, we notice that the Benasciutti estimate for the drive train lifetime is off by 19% compared to the simulation result.

Design example #2 This example is identical to the previous example except that we now allow the variability of the produced power to reach 0.05 instead of 0.01. The optimisation results shows, though, that even though lifetimes are increased a bit compared to the previous example, the extra overhead is not completely exchanged into increased lifetime. The variability of the produced power is 0.0136. That is, the power variability constraint is not active at the solution. This indicates that there is an upper limit for the damage rate reduction that can be achieved by relaxing the power quality demand.

Design example #3 This design constitutes a rare example of solidarity in control engineering. The increased tower base diameter would, if no changes in the controller were made, increase the tower lifetime by a factor of approximately $1.46^k = 4.6$. By applying the controller optimisation, though, this lifetime prolongation is shared among the components to reach an overall lifetime prolongation of approximately 25% compared to design example #1. The trajectories in figure 7.1 illustrates that the nacelle position has larger fluctuations, freeing the pitch system and the torque control from their tower damping duties.

An interesting feature of the resulting drive train deformation is, that the amplitude of the deformation is not significantly lowered. The fatigue damage reduction is achieved through a reduction of the high-frequency contents in the signal. That is, the number of damage inflicting stress cycles per time unit is lowered, hereby reducing the damage rate.

Baseline controller			
	Estimate	Simulated	Estimate off by
L_θ	100.00	100.33	0 %
L_z	100.00	106.28	6 %
L_β	100.00	99.91	0 %
c_{v,ω_g}	0.0025	0.0024	-1 %
c_{v,P_e}	0.0023	0.0023	-1 %
Design example # 1			
	Estimate	Simulated	Estimate off by
L_θ	211.26	260.47	19 %
L_z	211.30	219.99	4 %
L_β	211.30	210.96	0 %
c_{v,ω_g}	0.0100	0.0098	-2 %
c_{v,P_e}	0.0097	0.0097	0 %
Design example # 2			
	Estimate	Simulated	Estimate off by
L_θ	226.78	248.40	9 %
L_z	226.82	242.81	7 %
L_β	226.82	226.47	0 %
c_{v,ω_g}	0.0100	0.0098	-2 %
c_{v,P_e}	0.0136	0.0133	-2 %
Design example # 3			
	Estimate	Simulated	Estimate off by
L_θ	264.84	232.98	-14 %
L_z	264.86	280.26	5 %
L_β	264.86	263.37	-1 %
c_{v,ω_g}	0.0100	0.0097	-4 %
c_{v,P_e}	0.0100	0.0097	-3 %

Table 7.1: Design results.

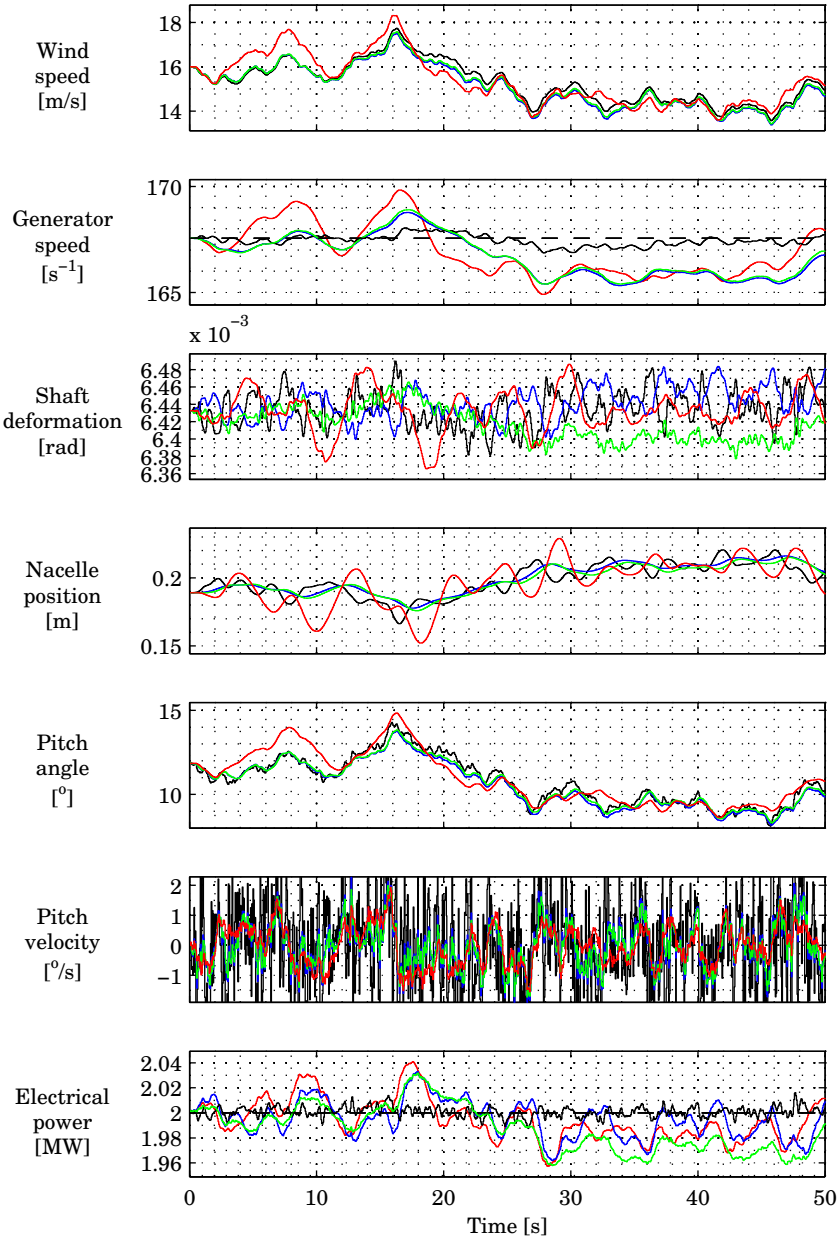

Figure 7.1: Optimisation of state feedback-controller. Black: Baseline controller. Blue: Design example #1. Green: Design example #2. Red: Design example #3

7.7 A few remarks on the optimisation problem

7.7.1 Ensuring reliable results for unstable iterants

With the gain matrix L being the decision variable, there is a risk that the optimisation routine applied to the problem would compute the cost function and the constraint functions for iterants resulting in an unstable closed-loop system. The variance of an unstable system should, ideally, be computed as infinite. Using the techniques for variance computation described in the preceeding sections will, unfortunately, result in negative variances when used on unstable system models.

In the present work, this has not shown to be a problem, as the steps taken by the optimisation algorithm are small enough to avoid jumping directly into the unstable area. A robust implementation would have to include countermeasures against this phenomenon.

7.7.2 Convexity

The optimisation problem (7.1) defines the solution as the global minimum of the cost function. The optimisation routine `fminimax` returns a local minimum. In effect, a global minimum at the solution found by `fminimax` is assured only if the objective function is convex.

Determining whether the cost function is convex in the controller gain matrix is not trivial. We will, for the time being, assume that the starting point given by the baseline controller is close enough to the global minimum to allow for practically useful results to be obtained.

It should be noted that techniques for global optimisation do exist. However, a treatment is considered out of scope for this work.

Chapter 8

Conclusions

8.1 Results

The work described in chapters 2 and 3 illustrates the need for non-trivial control schemes for wind turbines. The problem of maximising produced power, subject to constraints on the generator speed, led to the concept of operation modes, which, in turn, necessitated a hybrid scheme with different controllers for each mode of operation. Further, the inherent non-linearities stemming from the aerodynamic model necessitated a gain-scheduled scheme for mode IV-operation.

Also, in chapter 3 some of the trade-offs when tuning wind turbine controllers were demonstrated, as an LQI control scheme was tuned to meet a demand for damping of the tower oscillations. The price paid for this damping was shown to be increased pitch activity as well as increased variability of the generator speed and the produced power.

Following the treatment of time-domain fatigue analysis and spectrally based fatigue estimation in chapters 4 and 5, a major result was presented in chapter 6.

At first glance, the definition of spectral moments indicates that computation of

spectral moments would have to include a numerical integration of the spectral density function. Investigation of the symmetry governing the rational spectra of linear systems revealed, though, that general results concerning the integration of rational functions could be applied in a very convenient manner for the definite integral defining the spectral moments. In turn, an algorithm was developed that efficiently computes the spectral moments of linear, stochastic processes.

The major benefit of this technique is, that fatigue damage can be estimated *without any time-domain simulation*, and without running the sequential four-point algorithm on extensive simulation results. Thus, the computational work is reduced dramatically.

The combination of the spectral moment algorithm and the Benasciutti approximation provides the means for efficient evaluation of a cost function described in terms of fatigue damage rates for a linear closed-loop system. In chapter 7, controller optimisation was formulated as a constrained minimax problem with the damage rates for the tower, drive train, and pitch bearings making up the vector-valued cost function. The problem was constrained by upper limits for the variability of the generator speed and the produced power. Further, the problem was restricted to the class of state-feedback controllers, leaving the controller gain matrix as the decision variable.

Three controller design problems were worked out, from which we note the following results:

- The optimisation problem was, with the starting point being a stable textbook controller design, a well-posed problem.
- The optimisation algorithm succeeded in finding the proper trade-offs such that the damage rates of the three components in play were levelled to yield to highest possible minimum lifetime.
- Increased strength in one component was traded for longer lifetimes in the other components to provide equal lifetimes for all components.

8.2 Suggestions for further work

The framework established in this work has some obvious shortcomings that motivate further development.

- The wind turbine model should be extended to include the rotor dynamics.

In particular, a model for the blade root moment would eliminate the need for the crude approximation of constant blade root moment used in the pitch bearings fatigue estimation.

- In the modelling described in chapter 2, pure time delays were omitted from the actuator models. Since time delays in general tend to cause problems regarding stability, including these time delays in the model by e.g. Padé-approximations, might impose severe limitations on the achievable control performance.

- A shortcut has been made in this project regarding state information. The state feedback controller setup used in the optimisation demonstration assumed perfect state information. In a practical setup, a state estimator would have to be included in the setup as well as sensor models. The latter might impose severe constraints, depending on the availability and performance of sensors on the actual wind turbine.

- The validity of the presented methods should be confirmed by comparison to detailed, aeroelastic simulations followed by state-of-the-art fatigue estimation methods.

In addition, the work leaves a couple of more theoretical issues unsolved.

First, a theoretical proof of (6.6) is needed. Next, further theoretical work on the exploitation of symmetry properties of rational spectra when performing the integration described in section 6.1 might lead to an even more compact evaluation of some of the spectral moments.

Also, investigations of the accuracy of Benasciutti's approximation would be desirable, as one of the simulation results in section 7.6 revealed a Benasciutti estimate that was off by 19% compared to the simulated damage rate.

Finally, a major drawback in the analysis presented in this work is the restriction to continuous time descriptions. The development of a fatigue estimation framework formulated in discrete-time would greatly increase the applicability to real-world wind turbine controller design. Here, one of the major challenges lies in the interpretation of spectral moments when dealing with discrete-time spectra, and their application to a spectrally based fatigue estimate.

8.3 Perspectives

Rapid evaluation of damage rates in a linear system has been demonstrated through numerical optimisation of a wind turbine controller. It should be stressed, though, that the method presented is valid for *any* linear model driven by a Gaussian process. In effect, the use of the framework outlined in this work is not restricted to controller design. Performing optimisation of pure mechanical structures described by linear models is a possible application of the methods as well. This might prove useful in an initial design phase where a large number of structural models could be evaluated in terms of fatigue damage in short time.

Finally, it is worth revisiting the design examples given in section 7.6. A very important notion is, that the design objectives are related directly to management-level questions like "What are the costs of replacing the pitch bearings on site X?", "What are the power quality demands in country Y", or "Can the generator withstand a larger r.p.m. variability?".

In effect, we will conclude the thesis by stating that the main result of this work is the advent of these *management-level tuning knobs*, allowing controller tuning directly in terms of lifetimes. That is, a wind turbine design tool directly related to the economic trade-offs outlined in the introduction.

Appendix A

Hybrid controller design and implementation

A.1 The first order wind turbine model

With the drive train considered rigid, the equations of motion for the rotating parts can be stated as follows:

$$\dot{\omega}_r \left(I_r + N_g I_g\right) = T_a - \omega_r B_r - N_g \left(N_g \omega_r B_g + T_g\right) \tag{A.1}$$

Assuming an ideally controlled generator:

$$T_g = \frac{P_{\text{ref}}}{\omega_g} = \frac{P_{\text{ref}}}{N_g \omega_r},$$

and using the relationship $P_a = T_a \omega_r$ for the rotor, (A.1) becomes:

$$\dot{\omega}_r I_t = \frac{P_a}{\omega_r} - \omega_r \left(B_r + N_g^2 B_g\right) - \frac{P_{\text{ref}}}{\omega_r}, \tag{A.2}$$

where I_t denotes the "lumped" intertia:

$$I_t = I_r + N_g^2 I_g$$

Further assuming a rigid tower and ideal pitch actuators:

$$\dot{z} = \ddot{z} = 0 \quad , \quad \beta = \beta_{\text{ref}},$$

equation (A.2) completely describes the dynamics in a 1st order diminished wind turbine model. A linearised description (see section 2.10) in the deviation variables $\tilde{\omega}_r$, $\tilde{u}$, and $\tilde{y}$ around an equilibrium point $\langle \bar{\omega}_r, \bar{u}, \bar{y} \rangle$ is found as:[1]

$$\dot{\tilde{\omega}}_r = A^\circ \tilde{\omega}_r + B^\circ \tilde{u} \tag{A.3a}$$
$$\tilde{y} = C^\circ \tilde{\omega}_r + D^\circ \tilde{u} \tag{A.3b}$$

where

$$A^\circ = \frac{\partial \dot{\omega}_r}{\partial \omega_r} = \frac{1}{I_t} \left\{ -\frac{1}{\bar{\omega}_r^2} \frac{1}{2} \rho \pi R^2 \bar{v}^3 C_P \left(\bar{\lambda}, \bar{\beta}_{\text{ref}} \right) + \frac{1}{\bar{\omega}_r} \frac{1}{2} \rho \pi R^3 \bar{v}^2 \frac{\partial C_P \left(\bar{\lambda}, \bar{\beta}_{\text{ref}} \right)}{\partial \lambda} + \frac{P_{\text{ref}}}{\bar{\omega}_r^2} - \left(B_r + N_g^2 B_g \right) \right\}$$

and

$$B^\circ = \begin{pmatrix} \frac{\partial \dot{\omega}_r}{\partial \beta_{\text{ref}}} & \frac{\partial \dot{\omega}_r}{\partial P_{\text{ref}}} \end{pmatrix} = \begin{pmatrix} \frac{1}{\bar{\omega}_r} \frac{1}{2} \rho \pi R^2 \bar{v}^3 \frac{\partial C_P \left(\bar{\lambda}, \bar{\beta} \right)}{\partial \beta} \\ \frac{1}{\bar{\omega}_r^2} \end{pmatrix}$$

$$C^\circ = \begin{pmatrix} \frac{\partial \omega_g}{\partial \omega_r} \\ \frac{\partial P_e}{\partial \omega_r} \end{pmatrix} = \begin{pmatrix} N_g \\ 0 \end{pmatrix}$$

$$D^\circ = \begin{pmatrix} \frac{\partial \omega_g}{\partial \beta_{\text{ref}}} & \frac{\partial \omega_g}{\partial P_{\text{ref}}} \\ \frac{\partial P_e}{\partial \beta_{\text{ref}}} & \frac{\partial P_e}{\partial P_{\text{ref}}} \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}.$$

Note that the system state vector is reduced to the scalar ω_r. Further, note that the diminished model contains a direct term from P_{ref} to P_e ($D^\circ \neq 0$) because of the assumption of an ideal generator.

A.2 PI controller design

This section outlines the PI controller design used for the hybrid controller in chapter 3.

First, recall that the SISO control configurations in figure 3.3 all define the generator speed ω_g as the input signal to the controller. That is, in the SISO setup, there is no direct term from the control input (β_{ref} or P_{ref}) to the output ω_g ($D = 0$). In effect, the discrete-time first-order models for the wind turbine

[1] The superscripted circle $^\circ$ denotes a *diminished* system, similar to the $^\circ$ denoting diminished chords in musical notation.

in the SISO setups all have the form

$$\begin{aligned} x_{i+1} &= f x_i + g u_i \\ y_i &= c x_i. \end{aligned}$$

Assuming a PI controller:

$$\begin{aligned} \hat{x}_{i+1} &= \hat{x}_i + y_i \\ u_i &= k_i \hat{x}_i + k_p y_i, \end{aligned}$$

the closed loop description becomes

$$\begin{pmatrix} x \\ \hat{x} \end{pmatrix}_{i+1} = \begin{pmatrix} f + g k_p c & g k_i \\ c & 1 \end{pmatrix} \begin{pmatrix} x \\ \hat{x} \end{pmatrix}_i . \tag{A.4}$$

The eigenvalues λ of the 2×2 system matrix are

$$\lambda = \frac{1}{2}\left(f + g k_p c + 1\right) \pm \sqrt{4 g k_i c + \left(f + g k_p c - 1\right)^2}. \tag{A.5}$$

Now, assuming complex conjugated eigenvalues:

$$\Re(\lambda) = \frac{1}{2}(f + g k_p c + 1) \tag{A.6}$$

$$\Im(\lambda) = -\frac{1}{2}\sqrt{4 g k_i c + \left(f + g k_p c - 1\right)^2} \tag{A.7}$$

which gives, for the controller parameters k_p and k_i:

$$k_p = \frac{2\Re(\lambda) - f + 1}{g c} \tag{A.8}$$

$$k_i = \frac{\Im(\lambda)^2 - \frac{1}{4}\left(f + g k_p c - 1\right)^2}{g c} \tag{A.9}$$

Thus, the controller parameters are given as functions of the real and imaginary parts of the desired z-plane pole locations.

A.3 Selected simulation results

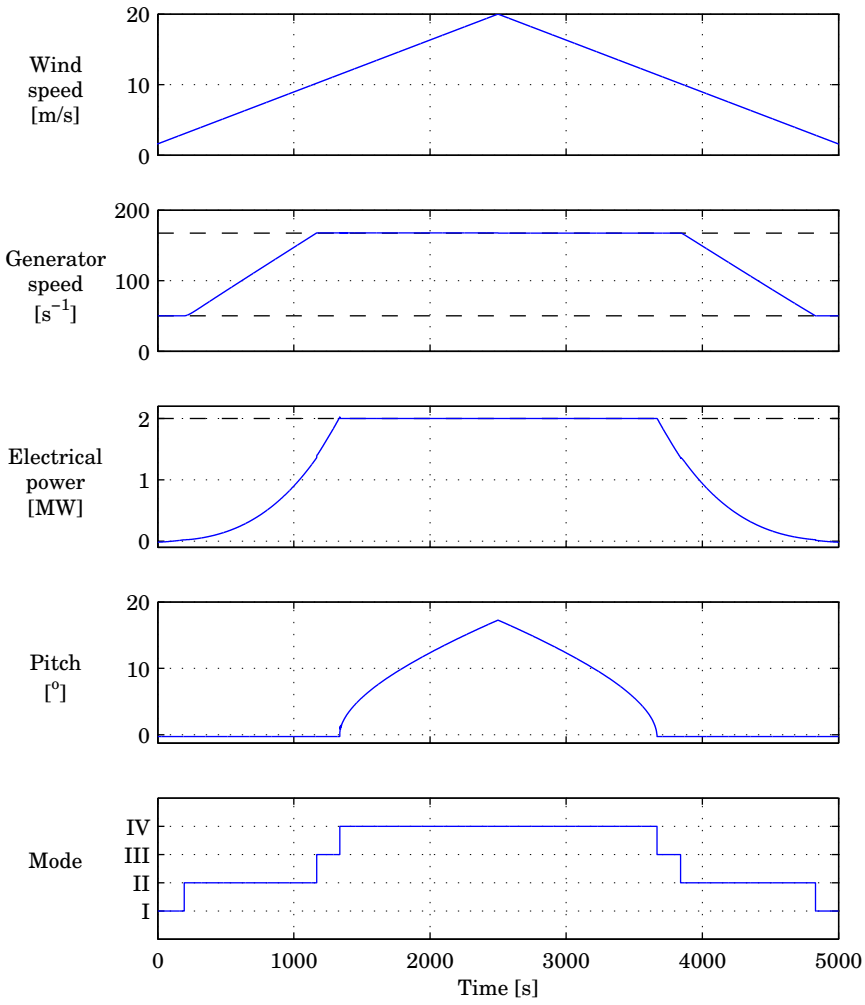

Figure A.1: Hybrid controller — quasi-stationary operation. Compare left part of figure with figure 3.2

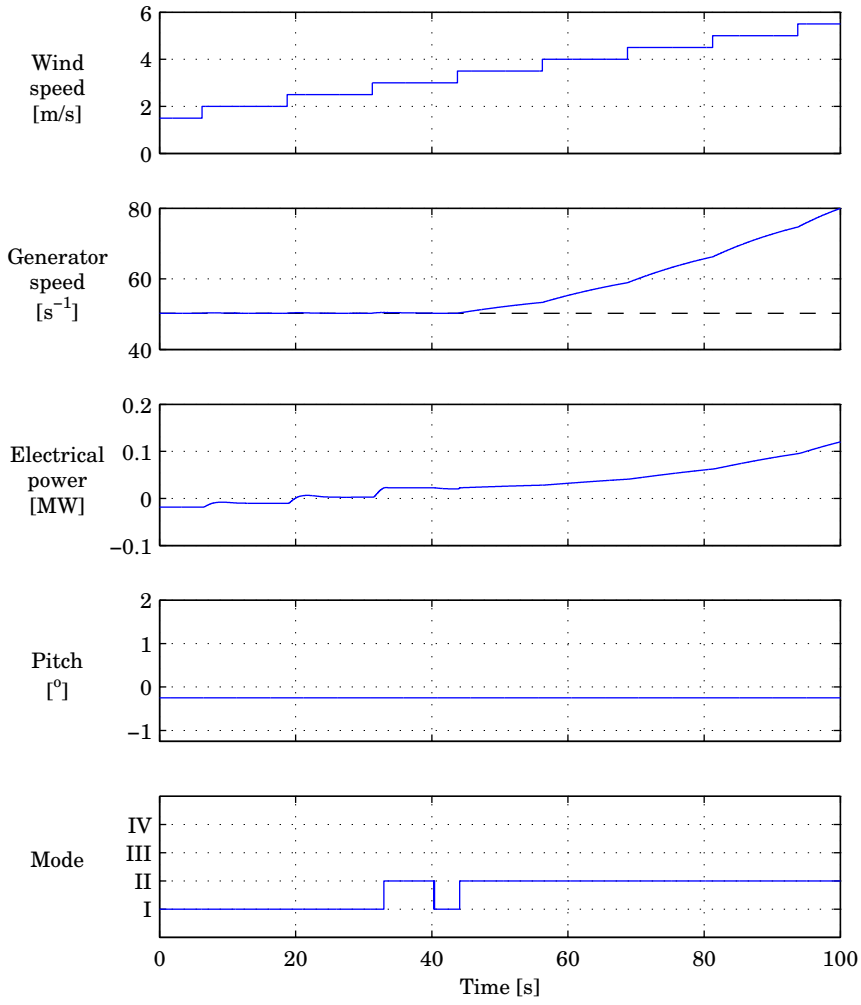

Figure A.2: Hybrid controller — step responses around mode I/II transition region. Note that mode toggles due to oscillating behaviour.

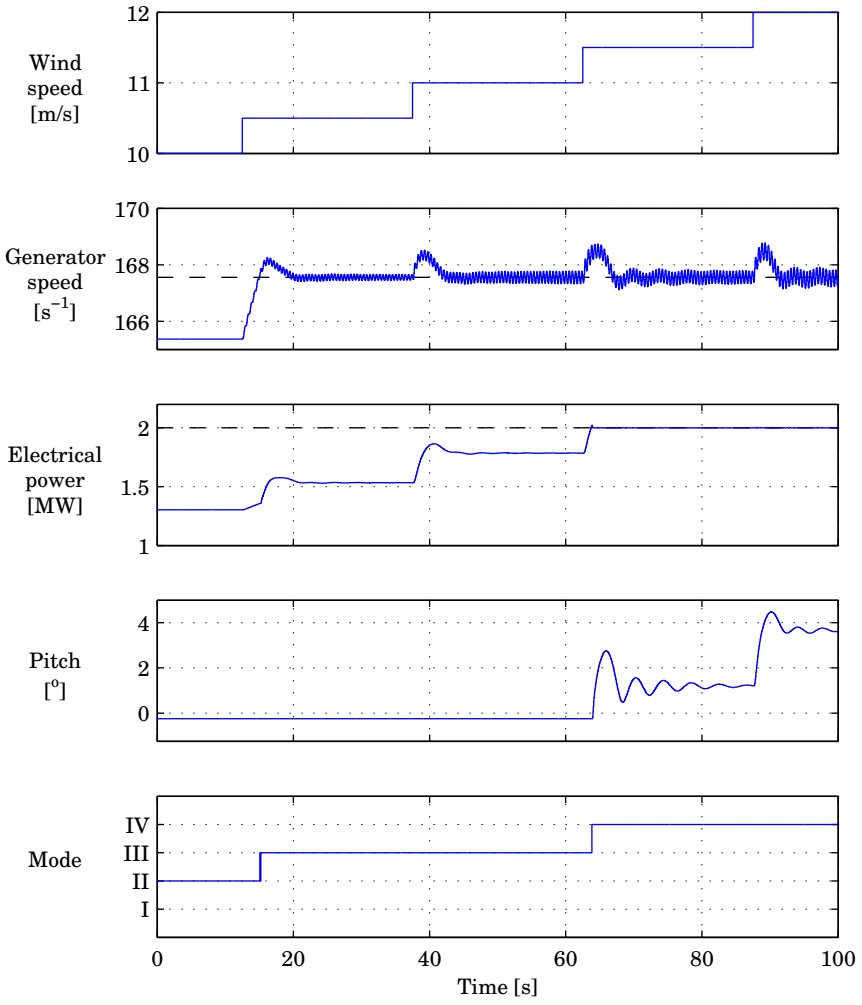

Figure A.3: Hybrid controller — step responses around mode II/III/IV transition region.

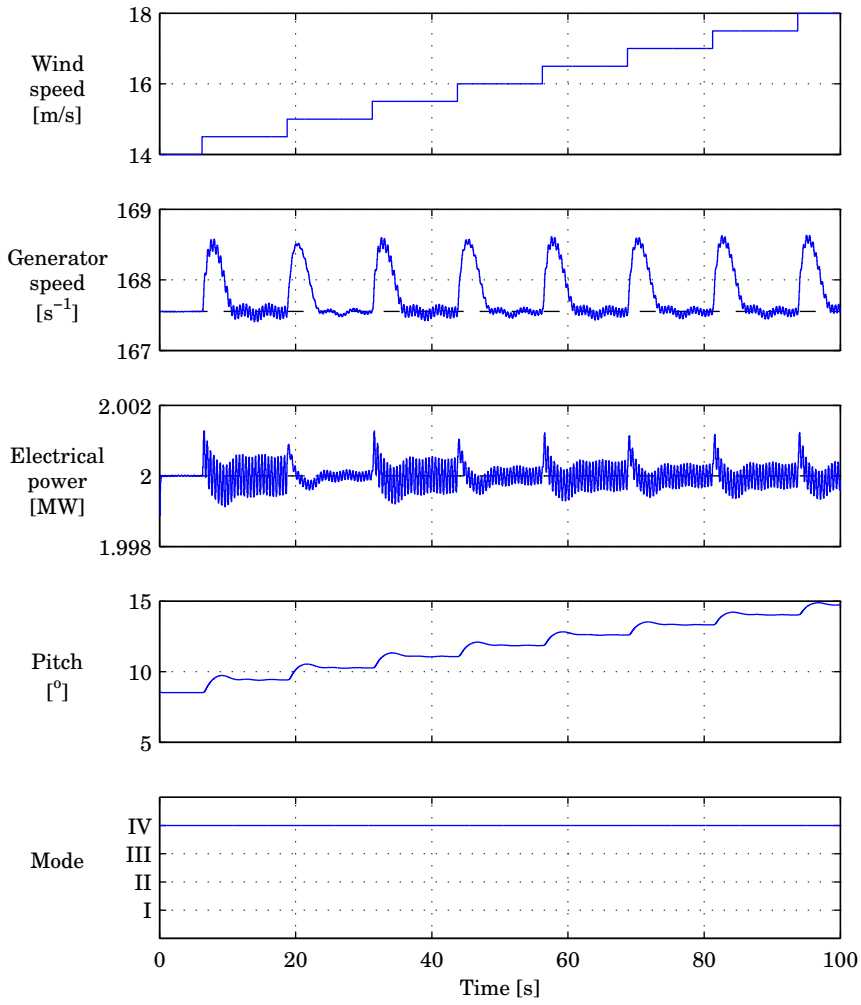

Figure A.4: Hybrid controller — step responses in mode IV. Note that due to gain scheduling the dynamic properties are more or less unchanged even though the operating point changes.

A.4 The hybrid controller function

```
function output = hybridcontroller(input)
% Discrete time hybrid controller
%
% Written August 2006 by Keld Hammerum

global wt;
global hybrid;

mode=input(1);
Pe=input(2);
omegag=input(3);

setpoint=[wt.omegagmin omegag wt.omegag0 wt.omegag0];

% Update error signal and accumulator
err=omegag-setpoint(mode);
hybrid.acc=hybrid.acc+err;

% Determine new mode

tol=0.01;

switch mode

    case 1
        if hybrid.Pr>wt.Plow, mode=2; end
    case 2
        if omegag<wt.omegagmin
            mode=1;
            hybrid.acc=(hybrid.Pr-hybrid.kp(mode)*err)/hybrid.ki(mode);
        end
        if omegag>wt.omegag0
            mode=3;
            hybrid.acc=(hybrid.Pr-hybrid.kp(mode)*err)/hybrid.ki(mode);
        end
    case 3
        if hybrid.Pr<wt.Phigh, mode=2; end
        if hybrid.Pr>wt.P0*(1+tol)
            mode=4;
            [kki,kkp]=getgains(wt.aero.beta_star);
            hybrid.ki(4)=kki;
            hybrid.kp(4)=kkp;
            hybrid.acc=(hybrid.betar-hybrid.kp(4)*err)/hybrid.ki(4);
        end
    case 4
        if (hybrid.betar<wt.aero.beta_star)&&(omegag<wt.omegag0)
            mode=3;
            hybrid.acc=(hybrid.Pr-hybrid.kp(mode)*err)/hybrid.ki(mode);
        end
end

% Compute new output signals

switch mode

    case 1
        hybrid.Pr=hybrid.ki(mode)*hybrid.acc+hybrid.kp(mode)*err;
        hybrid.betar=wt.aero.beta_star;
    case 2
        hybrid.Pr=Pomega(omegag);
        hybrid.betar=wt.aero.beta_star;
    case 3
```

```
        hybrid.Pr=hybrid.ki(mode)*hybrid.acc+hybrid.kp(mode)*err;
        hybrid.betar=wt.aero.beta_star;
    case 4
        hybrid.Pr=wt.P0;
        hybrid.betar=hybrid.ki(4)*hybrid.acc+hybrid.kp(4)*err;
        hybrid.betar=max(hybrid.betar,min(wt.aero.beta));
        % Update gains
        [kki,kkp]=getgains(hybrid.betar);
        hybrid.ki(4)=kki;
        hybrid.kp(4)=kkp;
        % Adjust accumulator
        hybrid.acc=(hybrid.betar-hybrid.kp(4)*err)/hybrid.ki(4);
end

% Form output vector
output(1)=mode;
output(2)=hybrid.Pr;
output(3)=hybrid.betar;

end
```

Appendix B

Selected source code listings

B.1 Damage computation - `damage.m`

```
function D = damage(x,k,K)
% Computes equivalent cycles s in x using rainflow count
% algorithm.
% Next, uses an analytical expression for the S-N curve
% to compute the inverse of the number of cycles that
% the material can withstand the stress amplitudes given in s
%
% % Written September 2006 by Keld Hammerum

s=rfc(x);
invN=(1/K)*s.^k;
D=sum(invN);
```

B.2 Computing spectral moments - `spectralmoments.m`

```
function lambdas=spectralmoments(z,p,k,var,which)
% Computes the spectral moments for the spectrum
% specified by H(jw)H(-jw)*var
%
% Input arguments
%
%    z        Vector of zeros in H(s)
%    p        Vector of poles in H(s)
%    k        Gain of H(s)
%    var      Variance (intensity) of driving noise
%    which    Specifies which moments to compute
%
% Ouputs
%
%    lambdas     Vector of length(which) containing the spectral moments
%
% Written October 2006 by Keld Hammerum

z=z(:);p=p(:);

% Create index vectors for real and complex zeros
zc=find(imag(z));zr=1:length(z);zr(zc)=0;zr=zr(find(zr));
% Create index vectors for real and complex poles
pc=find(imag(p));pr=1:length(p);pr(pc)=0;pr=pr(find(pr));

% Create polynomiums P and Q
P=1;Q=1;
% Add real zeros
for ii=1:length(zr)
    zz=z(zr(ii));
    P=conv(P,[1 0 abs(zz)^2]);
end
% Add complex zero pairs
for ii=1:2:length(zc)
    zz=z(zc(ii));
    P=conv(P,[1 0 2*(real(zz)^2-imag(zz)^2) 0 abs(zz)^4]);
end
% Add real poles
for ii=1:length(pr)
    pp=p(pr(ii));
    Q=conv(Q,[1 0 abs(pp)^2]);
end
% Add complex pole pairs
for ii=1:2:length(pc)
    pp=p(pc(ii));
    Q=conv(Q,[1 0 2*(real(pp)^2-imag(pp)^2) 0 abs(pp)^4]);
end

Qprime=polyder(Q);

% Compute integral
a=-j*p; % One half of the complex poles of Q(x)

nlambda=length(which);lambda=zeros(1,nlambda);

% Qprime is not changed therefore compute R2 once and for all
R2=polyval(Qprime,a);

for jj=1:nlambda
    % Create Pbar polynomium
    Pbar=[P zeros(1,which(jj))];
    % Check for the existence of the integral
```

```
    if length(Pbar)>=length(Q)-1
        lambdas(jj)=NaN;
    else
        R1=polyval(Pbar,a);
        R=R1./R2;
        S=-pi*imag(R)-(real(R).*log(abs(a).^2)+2*imag(R).*atan2(real(a),imag(
            a)));
        lambdas(jj)=var*k^2*sum(S)/pi;
    end
end
```

B.3 Benasciutti's approximation - benasciuti.m

```
function d=benasciutti(lambda,k,K)
% Computes Benasciutti's approximation to
% damage rate d. For the material with SN-curve
%
%     s^k N = K
%
% Input parameters
%
%    lambda    Vector of spectral moments:
%              [ m0 , m1 , m2 , m4 ]
%
% Written October 2006 by Keld Hammerum

m4=lambda(4); m2=lambda(3); m1=lambda(2); m0=lambda(1);

dnb=(1/(2*pi))*sqrt(m2/m0)*(1/K)*(2*sqrt(2*m0))^k*gamma(1+k/2);
a1=m1/sqrt(m0*m2);
a2=m2/sqrt(m0*m4);
b=(a1-a2)*(1.112*(1+a1*a2-(a1+a2))*exp(2.11*a2)+(a1-a2))/(a2-1)^2;
d=dnb*(b+(1-b)*a2^(k-1));
```

B.4 The four-point algorithm - `rfc.m`

```
function S = rfc(x)
% Computes the equivalent cycles for the time series in x
%
% % Written September 2006 by Keld Hammerum

x=x(:)';
% Convert to extremum sequence
extrema=toextremes(x);
% Extract all inner cycles
[S,res]=extractinnerranges(extrema);
% Handle residual. Cloormann/Seeger method as in WAFO. First
% ensure that [res res] only consists of extremum points
extrema=toextremes([res res]);
% Next, extract inner ranges
[Sres,resres]=extractinnerranges(extrema);
% Add the inner ranges from the residual to the stress range vector
S=[S Sres];
end


function [S,res]=extractinnerranges(extrema)
i=1;S=[];
while i<length(extrema)-2
    % Pick out four extremes
    E=extrema(i:i+3);
    % Check for inner range
    if (E(2)>E(1) && E(3)>=E(1) && E(4)>=E(2))||(E(2)<E(1) && E(3)<=E(1) && E
        (4)<=E(2));
        % Add inner range to stress range vector S
        S=[S abs(E(2)-E(3))];
        % Remove inner range from extrema
        extrema=[extrema(1:i) extrema(i+3:end)];
        % Move back to check if we have created another inner range
        i=max(1,i-2);
    else
        % Not an inner range - move forward
        i=i+1;
    end
end
res=extrema;
end


function y=toextremes(x)
diffs=diff(x);
x=[x(find(diffs~=0)) x(end)]; % Remove consecutive identical values
y=x(find(diff(sign(diff(x))))+1);
% Add ends
y=[x(1) y x(end)];
end
```

Bibliography

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

[BT05] D. Benasciutti and R. Tovo. Spectral methods for lifetime prediction under wide-band stationary random processes. *International Journal of Fatigue*, 27, 2005.

[CO04] Sune Baun Cristensen and Niels Peter Ottsen. Wind turbine control - a model predictive control approach. Master's thesis, Aalborg University . Institute of Electronic Systems, 2004.

[Gro00] The WAFO Group. *WAFO - a Matlab toolbox for Analysis of Random Waves and Loads*. Lund Institute of Technology, Centre for Mathematical Sciences, Mathematical Statistics, 2.0.02 edition, August 2000.

[HHL+05] M. H. Hansen, A. Hansen, T. J. Larsen, S. Øye, P. Sørensen, and P. Fuglsang. Control design for a pitch-regulated variable speed wind turbine. Technical Report Risø-R-1500, Risø National Laboratory, 2005.

[Knu83] Torben Knudsen. Regulering af vindmøller. Master's thesis, IMSOR, DTH, 1983.

[Ma97] Xin Ma. *Adaptive Extremum Control and Wind Turbine Control*. PhD thesis, Institute of Informatics and Mathematical Modelling, Technical University of Denmark, 1997.

[Mad99] Peter Hauge Madsen. Recommended practices for wind turbine testing – 3. fatigue loads. Technical report, Risø, 1999. Part of IEC 61400-13.

[MCC+] Amir S. Mikhail, Craig L. Christenson, Kevin L. Cousineau, William L Erdman, and William E Holley. Variable speed wind turbine generator. Australian Patent Office - patent no. 737762.

[New84] D. E. Newland. *An Introduction to Random Vibrations and Spectral Analysis, 2nd edition.* Longman Scientific Technical, 1984.

[Ryc87] Igor Rychlik. A new definition of the rainflow cycle counting method. *International journal of fatigue*, 9, 1987.

[Ryc93] Igor Rychlik. On the narrow-band approximation for expected fatigue damage. *Probabilistic Engineering Mechanics*, 8, 1993.

[SBD05] Frank Sherrat, N.W.M. Bishop, and Turan Dirlik. Predicting fatigue life from frequency-domain data: current methods - part a. Available from Engineering Integrity Society, www.e-i-s.org.uk, 2005.

[SM92] T. N. Subramaniam and Donald E. G. Malm. How to integrate rational functions. *The American Mathematical Monthly*, 99(8), 1992.

[Vel06] Dick Veldkamp. *Chances in Wind Energy - A probabilistic Approach to Wind Turbine Fatigue Design.* PhD thesis, Delft University Wind Energy Research Institute, 2006.