NTNU

Norwegian University of
Science and Technology

Analysis of Accelerometric Datasets for Wind Turbine Monitoring

Jon Henning Rolfseng

Marine Technology
Submission date: June 2016
Supervisor: Ingrid Bouwer Utne, IMT
Co-supervisor: Nicolas Lefebvre, IMT

Norwegian University of Science and Technology
Department of Marine Technology

Preface

The master's thesis accounts for 30 credits and is a mandatory part of the master of science (MSc) degree within marine technology at NTNU. It takes place in the 10^{th} and final semester and is what completes the study program. The students are free to choose a topic as long as it is anchored in the education plan and is relevant to the maritime industry. Within these restrictions, the topic is selected based on qualification and interest. Having specialized within marine systems design constitutes no direct link to wind turbines - nor to the condition monitoring of their components, which is the topic for this thesis. Nevertheless, the platform of natural sciences which is integral to the MSc for technological disciplines at NTNU serves as a door opener also for undertakings that is not strictly correlated to the chosen specialization.

More specifically, the thesis analyses accelerometric datasets with the intention of monitoring component condition. The vibration data is collected from a land-based windfarm operated by Statkraft - a leading company in renewable energy being headquartered in Oslo, Norway. The analysis has called for deployment of sophisticated statistical methods and their implementation in MATLAB®.

The idea was brought up by Maintech AS - a company whose core competencies are centered around operation and maintenance. The fact that the project has support in serious industry players makes it exciting and yet very useful in terms of incorporating industry know-how into the thesis.

The report is intended for personnel having similar background as the author. A line of communication emphasizing clear and explanatory argumentation is sought at best ability.

Trondheim, 2016-10-06

Jon Rolfseng

Acknowledgment

Writing this master's thesis has relied on the goodwill and support of several people and could not have been completed otherwise.

I would like to thank my supervisors, Ingrid Bouwer Utne and Nicolas Lefebvre for providing very professional guidance throughout the process of developing this master's thesis. With her extensive knowledge within operation & maintenance, Ingrid Bouwer Utne has assisted with valuable proposals regarding the overall structure and organization of the project. Nicolas Lefebvre, who possesses great expertise within data analysis and statistics, has demonstrated extraordinary efforts and patience throughout this process by willingly sharing his knowledge. My most sincere gratitude goes to both.

Furthermore, I would like to thank Jan Erik Salomonsen, principal engineer at Maintech AS, for enthusiastically sharing industry knowledge on operation and maintenance in the wind industry sector. These contributions have enabled several arguments in the final report. As a consequence of his dialogue with operators in the wind industry we were finally able to set up a collaboration with Statkraft. Moreover I want to thank you for proposing this topic, which has proven very interesting.

Finally a big thanks to wind advisor at Statkraft, Jørgen Togstad. Firstly for accepting to share sensor-data but also for arranging a pleasant visit at the company headquarters in Lysaker, Oslo. By providing us with access to the windfarm site-server and with information on how to maneuver in the database you made sure that we did not return empty handed.

J.Rolfseng

Summary

Compared to hydrocarbon alternatives, wind powered electricity comes at nearly twice the cost. With operation & maintenance accounting for 20-30% of the wind farm life cycle cost, the enabling of smarter maintenance is found paramount - particularly for offshore wind. With the intention of exploiting component life, modern turbines are extensively equipped with sensor and control systems, allowing for condition based maintenance. However, due to conflicting interests among stakeholders, the industry is characterized by lack of data sharing, preventing the field of intelligent fault diagnosis and prognosis from offering its full range of benefits.

This project has gained access to vibration data from a land-based Norwegian wind farm. As there is no information regarding system/component status (fault log information), the following research questions becomes relevant: To what extent is it possible to monitor the condition of the wind turbine gearbox using unlabeled vibration data? Partial objective (PO) 1 and 2 underpin the main objective of the thesis, namely to develop and apply an approach for condition monitoring for the wind turbine gearbox.

PO 1 Segregate vibrations caused by load from those being attributed to degradation.

PO 2 Relate the vibration-induced response to time.

The work is initially carried out based on the below hypotheses. The issue of missing information is largely dealt with by assumption I.

Hypothesis I Most of the data correspond to normal turbine behavior.

Hypothesis II All turbines can be considered equal, i.e., they give rise to the same vibration response under equivalent conditions.

Hypothesis III The vibration response can be modeled as a function of operational load and degradation level in an additive manner.

Hypothesis III is effectuated by assuming that the vibration amplitude, y_f, is composed of operating conditions oc (rated power, wind speed etc.) and degradation d (degradation):

$$y_f \sim f(oc, d) \tag{1}$$

where oc and d are assumed to be additive contributions to the vibration amplitude:

$$y_f \sim f_1(oc) + g(d) \tag{2}$$

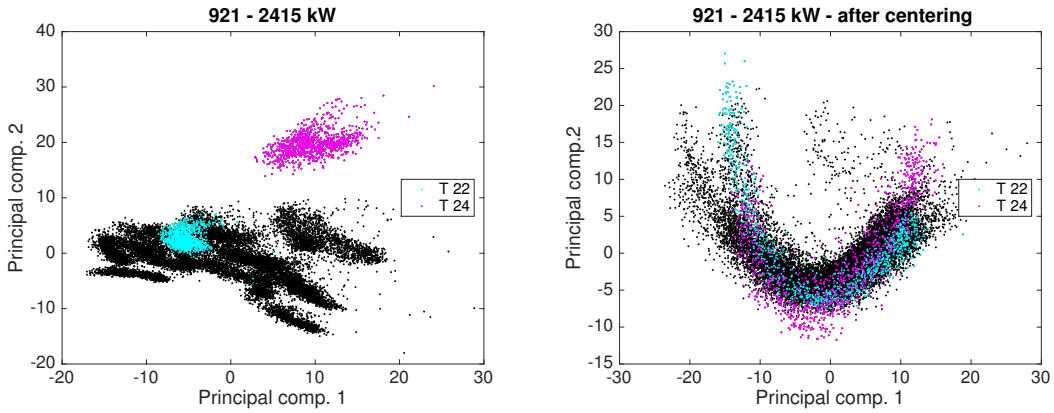
By deploying a model estimating the part of y_f being a result from oc, the presumed fraction being attributed to d is approximated through the residual, Δf:

$$g(d) \sim \Delta f = \hat{f}_1(op) - f(oc, d) \tag{3}$$

The process of obtaining $g(d)$ is expressed through the following steps:

- Filtering/pre-processing → Input for model training
- Model training and evaluation → Generalization with 1^{st} - 3^{rd} order polynomials and through artificial neural networks (ANN). Best option is selected.
- Residual analysis → Monitoring trends for vibrations being caused by degradation

Visualization through principal component analysis (fig 1) reveals that each turbine give rise to their own region of normal behavior, thus compromising assumption II. To enable multiple-turbine consideration, the offsets are corrected for by median-centering. Although turbine-specific behavior still can be recognized, the discrepancy is not considered sufficient to fully discard hypothesis II.



(a) No correction. **(b)** After Correction

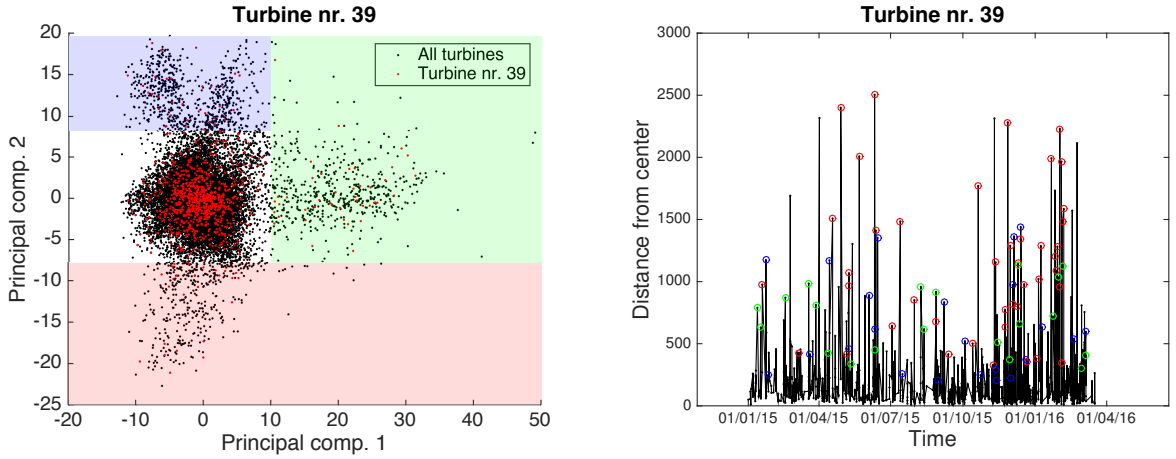
Figure 1: Left plot shows clusters before the data set is adjusted for turbine-specific behavior. Right plot depicts the resulting responses after the correction is made.

Among the model candidates the 20-neuron ANN makes the fest fit and is hence chosen. Based on their correlation with vibration amplitude, *rated power* and *wind speed* are chosen as model input, thus representing the *operating conditions*.

Table 1: Model performance - mean squared error.

Error	Linear	2^{nd} degree polynomial	3^{rd} degree Polynomial	Neural Net (10 neuron)	Neural Net (20 neuron)
MSE learning	0.00855	0.00713	0.00617	0.00562	0.00567
MSE testing	0.00856	0.00714	0.00618	0.00564	0.00558

The effect caused by the operating conditions is successfully removed by the ANN model. The below clusters (fig. 2, left) are thus not related to rated power or wind speed. Turbine 39, which is representative for the remaining turbines, form four distinct clusters representing the vibration response. To monitor the gearbox condition, the distance from the assumed normal behavior cluster (white background) is related to time (fig. 2, right). The rapid transitions between states do not show consistency with degradation. Whether this is the correct assessment, i.e., that no degradation has been present in the gearboxes, or whether the approach fails to detect it can not be given a definite answer. In case of the latter, inadequacy of assumption II is found a plausible explanation. Furthermore, the model may suffer from an inadequate set of input-parameters that do not fully cover all impacting variables. In that case, the residuals will represent also other phenomena than degradation.



(a) Three unlabeled zones outside "normal"

(b) Relating presence in each state to time

Figure 2: The rapid transitions between different clusters is not consistent with the degradation process.

Sammendrag

Sammenliknet med hydrokarboner er vind-drevet elkraftproduksjon cirka dobbelt så dyrt i gjennomsnitt. Utgifter til drift og vedlikehold utgjør 20-30 % av livssykluskostnaden for en vindpark og er derfor utpekt som fokusområde for økt lønnsomhet. For å muliggjøre tilstandsbasert vedlikehold er moderne vindturbiner utstyrt med et mangfold av måleutstyr. Imidlertid er vindindustrien kjennetegnet av tilbakeholdenhet når det gjelder deling av tilstandsdata. Dette er med å bremse forskning og utvikling som søker bedre utnyttelse av måledata.

I dette prosjektet brukes det vibrasjonsdata fra en norsk landbasert vindpark. At det ikke har vært mulig å fremskaffe feillogger som kan bekrefte tilstand for komponentene gjør følgende spørsmål relevant: I hvilken grad er det mulig å foreta tilstandsovervåking for girkassen i en vindturbin uten informasjon om komponentstatus? Delmål 1 og 2 understøtter hovedmålsettingen til oppgaven, nemlig utvikling og anvendelse av en tilnærming for tilstandsovervåking av girkassen i en vindturbin.

Delmål 1 Isolere bidrag til vibrasjonsrespons som skyldes degradering.

Delmål 2 Overvåke vibrasjon skapt av slitasje som funksjon av tid.

Arbeidet tar utgangspunkt i hypotesene nedenfor. Hypotese I er essensiell i håndtering av manglende informasjon.

Hypotese I Stordelen av dataen svarer til normal turbinoppførsel.

Hypotese II Turbinene kan betraktes like, altså vil turbiner som er utsatt for likt slitasjenivå, og som opererer under samme operasjonsforhold, gi opphav til tilsvarende vibrasjonsrespons.

Hypotese III Vibrasjonsresponsen kan modelleres som et additivt bidrag utgjort av operasjonslast og degraderingsnivå.

Vibrasjonsamplituden, y_f, antas å være en funksjon av operasjonsforhold, of, (effekt, vindhastighet, orientering etc.) og degradering, d:

$$y_f \sim f(of, d) \tag{4}$$

hvor of og d er antatt å gi additive bidrag til vibrasjonsresponsen:

$$y_f \sim f_1(of) + g(d) \tag{5}$$

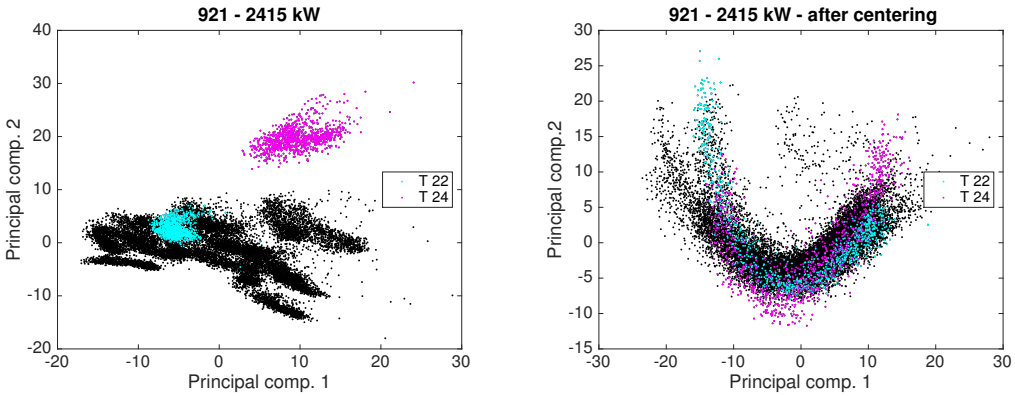
Gjennom en modell som estimerer det bidrag til y_f som er et resultat av of, vil bidraget som skyldes degradering, d, kunne estimeres fra forskjellen mellom modell-output og faktisk måling:

$$g(d) \approx \Delta f = \hat{f}_1(of) - f(of, d) \tag{6}$$

Metode/tilnærming for estimering av $g(d)$ er gitt av punktene under:

- Filtrering/før-prosessering → Input til modellering.
- Modellering og vurdering av ytelse → Generalisering med polynomer av 1., 2. og 3. grad og med kunstige nervenettverk. Velger modell som yter best.
- Analyse av $g(d)$ → Overvåke trend/utvikling av vibrasjoner som er en følge av degradering.

I før-prosesseringen avdekkes det at hver turbin gir opphav til spesifikke klynger som representerer vibrasjonsrespons - altså svekkes holdbarheten i hypotese II. For å muliggjøre samtidig analyse av flere turbiner, korrigeres dette for gjennom median-sentrering. Figur 3 viser spesifikk respons fra turbin 22 og 24 før/etter bias-korrigering. Selv om turbinresponsen ikke er sammenfallende etter korrigering (høyre) antas de å være tilstrekkelige like - altså bevares hypotese II.



(a) Før korrigering. **(b)** Etter korrigering

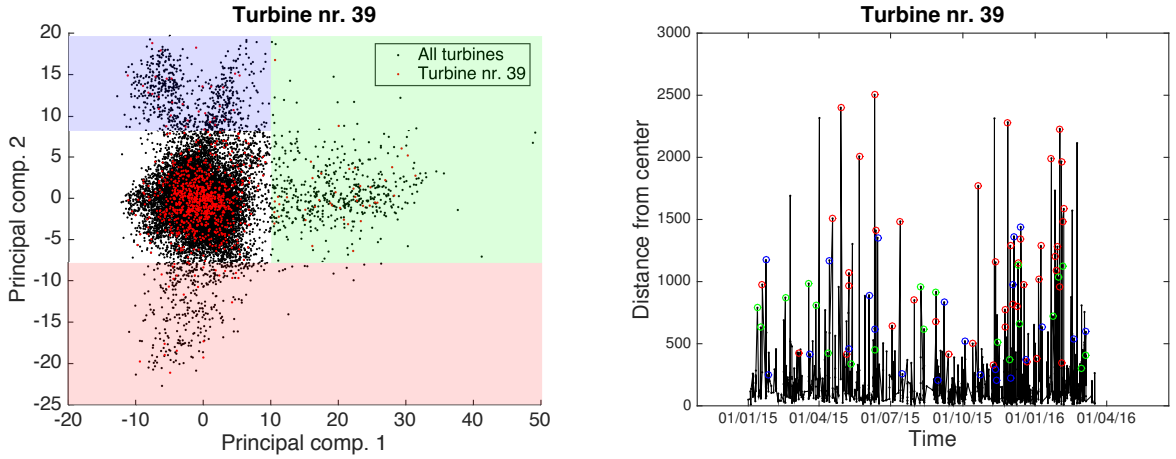
Figure 3: Venstre plott viser klynger før data-settet justeres for turbinspesifikk oppførsel. Høyre plott viser resulterende vibrasjonsrespons etter median-korrigering.

Blant modellene som vurderes, kommer det kunstige nervenettverket med 20 nevroner best ut og blir derfor valgt. Ut fra påvist korrelasjon mellom vibrasjonsamplitude og operasjonsforholdende *effekt* og *vindhastighet* velges disse som input for modelltrening.

Table 2: Ytelse av modeller målt med middelkvadratfeil.

Feil	**Lineær**	**2. ordens polynom**	**3. ordens polynom**	**nervenettverk (10 nevroner)**	**nervenettverk (20 nevroner)**
Feil, læring	0.00855	0.00713	0.00617	0.00562	0.00567
Feil, testing	0.00856	0.00714	0.00618	0.00564	0.00558

Nervenettverkmodellen er i stand til å fjerne vibrasjonsbidrag som skyldes effekt og vindhastighet. Klyngene nedenfor (fig. 4, venstre) er dermed ikke relatert til disse forhold. Turbin 39, som er representativ for resterende turbiner, former fire distinktive klynger for vibrasjonsrespons. For å indikere girkassens tilstand, overvåkes distansen fra klyngen som er antatt å representere normal oppførsel (hvit bakgrunn) som funksjon av tiden (fig. 4, høyre). Med hyppige og uregelmessige overganger mellom klynger kan det ikke påvises degradering. Hvorvidt dette er en riktig vurdering, altså at ingen slitasje har funnet sted, kan ikke sies med sikkerhet. Hvis girkassen på den annen side har opplevd slitasje, har metoden feilet i å oppdage det. Hvis så er tilfelle nevnes uriktigheten av hypotese II som mulig forklaring. Videre kan det være at effekt og vindhastighet ikke tilstrekkelig beskriver settet av input som styrer vibrasjon. I så fall vil responsen i figur 4 (venstre) representere utslag fra også andre forhold enn degradering.



(a) Three unlabeled zones outside "normal"

(b) Relating presence in each state to time

Figure 4: Uregelmessige og hyppige overganger mellom klynger gjør at degradering ikke kan påvises.

Contents

Chapter 1

Introduction

This chapter describes background and current challenges within the wind industry to form a rationale for the research objectives and delimitations considered in the thesis. These are presented at the end of this chapter along with the structure of the thesis. Although being modified and updated according to recent information, parts of the chapter overlap with Rolfseng (2015) (ch 1&2) where similar background information was required.

1.1 General Background

Both land-based and offshore wind has experienced dramatic growth during the recent years. In terms of global annual installed capacity, the 50 GW mark was exceeded for the first time in 2014 (GWEC, 2014) - a figure which according to the same report is predicted to continue with a 3 % - 6 % growth rate in the coming five years. Cumulative installed wind capacity within the last two decades is shown in figure 1.1.

When asking why this is happening, several answers arise. Wind energy emerge as one out of several possible instruments intending to mitigate greenhouse gas (GHG) emissions. The distribution of different energy sources as in the year of 2012 is depicted in figure 1.2. By early 2015, as many as 164 countries had defined renewable energy targets, showing that indeed the commitment to renewable energy has a global scope (Kieffer and Couture, 2015). Further, and even more fundamentally, the world is facing a rapid growth in energy demand. This go hand in hand with the world's growth in population, which according to DESA (2015) is expected to

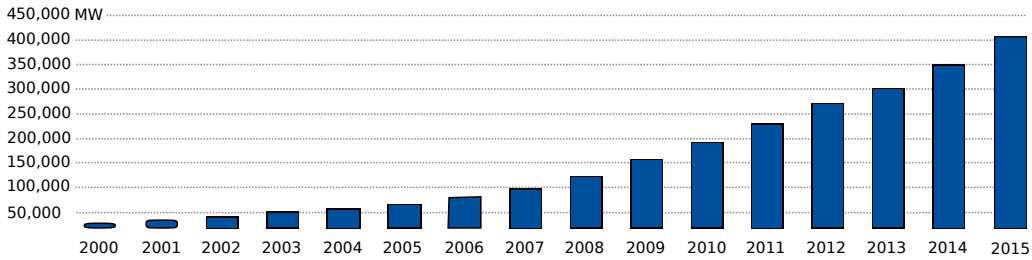


Figure 1.1: Global cumulative installed wind capacity during 2000-2015. Figure adapted from GWEC (2015).

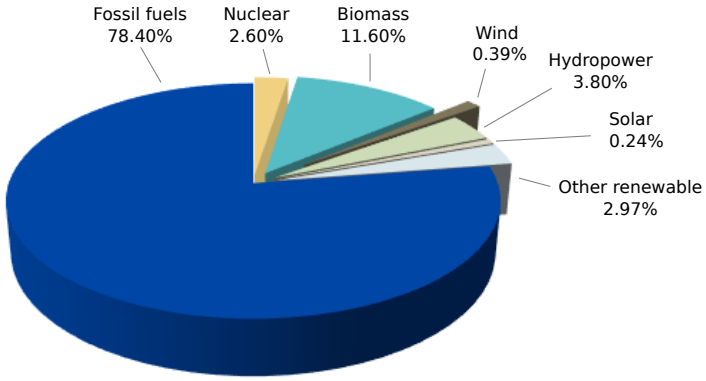


Figure 1.2: Shares of different energy sources in the total energy consumption in the year 2012. Other sources include: solar, geothermal and tidal energy sources. Figure adapted from Kumar et al. (2015).

exceed 9,7 billion by 2050. Considering the fact that fossil energy resources are finite resources, the shift towards renewable alternatives will at some point be inevitable. To date, the literature (e.g, see Kumar et al., 2015; GWEC, 2014; IEA, 2013) suggests that wind energy is one of the most promising renewable energy sources as it is stable and come at a relatively low operating cost. The stable energy supply is essentially made up by i) the availability of the wind resource, and ii) the availability of the technical system.

☞ **Availability for a wind turbine**: The ratio of the total number of hours during a certain period, excluding the number of hours that the wind turbine could not be operated due to maintenance or breakdown, to the total number of hours in the period (IEC, 1999).

Statements regarding reliable energy supply corresponds well for mature markets on land, which occasionally is very competitive already (Irena, 2012). Although relying on regular service

and fast response to fault situations (Dai, 2014), modern turbines on land generally can reach availability of 97-99% or more (IWES, 2014). This level of availability is not necessarily the case for the offshore wind farm (OWF). Older OFWs, typically deployed with relatively low capacity turbines close to shore, can sometimes exhibit availability in the range of the average onshore availability. However, as depicted in figure 1.3, the availability of more recently commissioned OWFs farms is less promising. This challenge is believed to grow even larger as the prevailing trend today is situating wind farms even farther ashore where weather is more hostile (e.g, see Ho et al., 2016; GWEC, 2014). With respect to maintenance, harsher weathering can be considered a two-dimensional issue, as i) the increased load will accelerate the degradation-processes and ii) because accessibility gets more susceptible.

☞ **Accessibility**: The percentage of time that the maintenance fleet can provide service to the offshore wind farm (Dai, 2014).

☞ **Maintenance**: The combination of all technical, administrative and managerial actions during the life cycle of an item intended to retain it in, or store it to, a state in which it can perform the required function (Marquez, 2007).

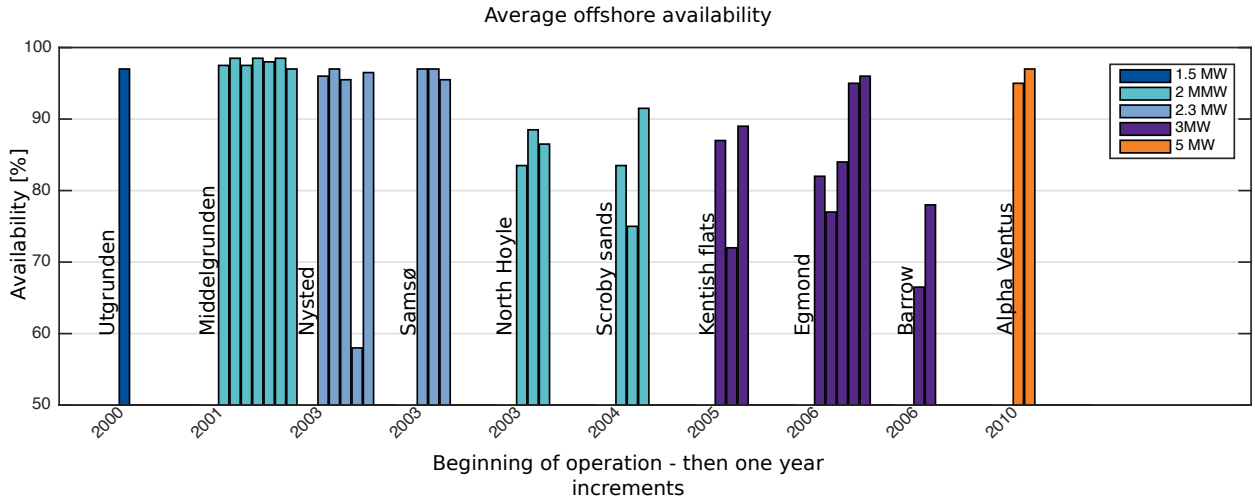


Figure 1.3: Average availability per year for a selection of European offshore wind farms. Figure adapted from IWES (2014).

Through their availability breakdown, van Bussel and Bierbooms (2003) provide a distinc-

tion between theoretical availability and actual wind farm availability. As seen in figure 1.4, the theoretical availability is largely composed design-related elements.

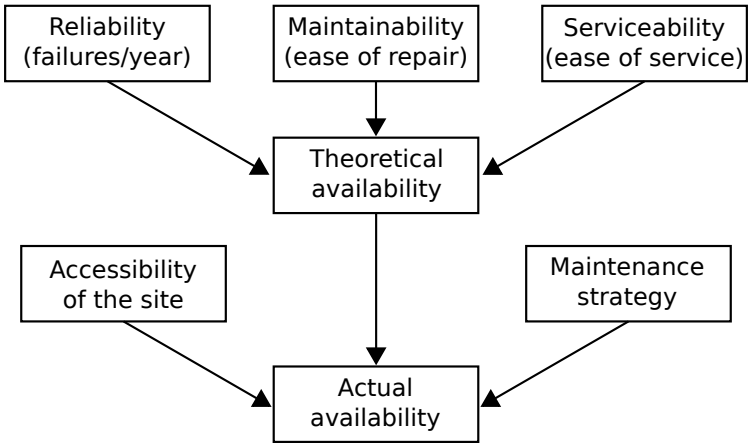


Figure 1.4: Theoretical and actual availability as a function of site accessibility, design properties and maintenance strategy. Figure adapted from van Bussel and Bierbooms (2003).

For the offshore segment where accessibility might be restricted several months out of a year (Dai, 2014), accessibility poses a major challenge in terms of aligning the maintenance strategy towards effective resource utilization. However, to make availability a productive metric in describing the effective operation of a wind farm, one must of course consider at which cost operational availability is achieved. When comparing wind-driven electricity with commonplace hydrocarbon energy sources, it still has a way to go in terms of being competitive.

As of 2013, the levelized cost of electricity (LCOE)[1] for offshore wind farms was roughly 200 USD/MWh on average, whereas the onshore farms was in the area of 130 USD/MWh. These quantities are found by Salvatore (2013), who compares the cost of electricity by source. For commonplace hydrocarbon energy sources, the LCOE tend to lie stably in the range from 50-100 USD/MWh. This makes an incentive for pursuing ways of reducing costs in the wind industry altogether, although particularly required for the offshore segment. With operation and maintenance (O&M) currently constituting 20 % - 35 % of the LCOE, it is highlighted a key issue for improving the viability of the offshore wind segment (Shafiee et al., 2015; Kumar et al., 2015; IEA, 2013).

The topic of reducing the operational expenditure for wind farms are not only a concern rele-

[1]The LCOE of a given technology is the ratio of lifetime costs to lifetime electricity generation, both of which are discounted back to a common year using a discount rate that reflects the average cost of capital (IRENA, 2015).

vant for owners and operators, but is anchored in a broader context intending to realize political renewable energy targets. Aligned with such visions is the NOWITECH project - an international research cooperation on offshore wind technology financed by the Research Council of Norway as well as their industry partners. [2] Among the areas of focus is the assessment and development of novel designs, but also a dedicated branch of research is directed towards O&M technologies and strategies.

The project has given rise to a plethora of spin-off projects, such as the LEANWIND project working more uniformly with O&M. To identify favorable utilization of maintenance resources, the project is supported by a wind farm lifecycle simulation model (Valland et al., 2014). A reoccurring topic has been whether a detailed degradation model is needed or not (Sperstad, 2015). As the simulation model intend to provide *strategic* rather than *tactical* and *operational* decision support it is believed that a high-level maintenance model will be satisfactory due to the long time horizons being considered (Sperstad, 2015).

With reference to figure 1.5 showing the hierarchical levels of the maintenance concept, the focus of this thesis will lie within the tactical level. Within this level, Pintelon and Parodi-Herz (2007) puts the maintenance policy. Their definition follows below:

☞ **Maintenance policy**: Rule of set of rules describing the triggering mechanism for the different maintenance actions.

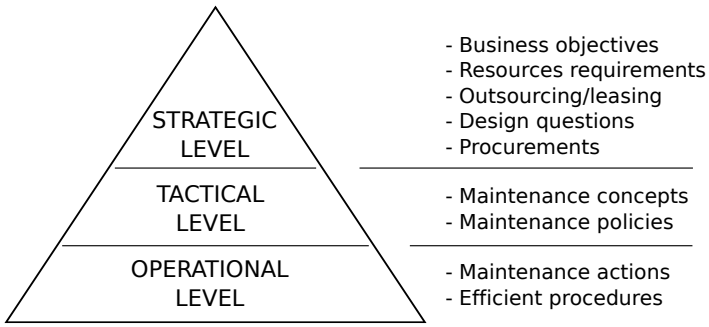


Figure 1.5: Maintenance levels.

The most common maintenance policies are run to failure (RTF), clock-based maintenance,

[2] https://www.sintef.no/projectweb/nowitech/

age-based maintenance, opportunity-based maintenance (OBM) and condition-based maintenance (CBM) - of which the latter is designated as most relevant for cost efficiency in the wind industry. Although condition monitoring already plays an important role in defining the triggering rule or criteria for when to initiate maintenance, the literature (e.g, see Tchakoua et al., 2014; Ata, 2015) agree that there is a large potential yet not exploited within the domain of CBM for the wind industry. This involves intelligent algorithms which by providing system health estimates and prognosis for remaining useful life enable for cost efficient utilization of maintenance resources and component life. These algorithms relies heavily upon access to data, a topic which currently poses a major challenge in terms of realizing smarter maintenance in wind farm operation.

1.2 Wind Turbine Data Generation

Modern wind turbines are equipped with extensive sensor-configurations and sophisticated controller systems - both capable of recording data which can be used for health monitoring purposes. Among the large variety of sensing and measuring techniques that are available for wind turbines, vibration analysis is the most predominant (Nie and Wang, 2013; Cornelius, 2004).

It is distinguished between i) dedicated condition monitoring (CM) equipment and ii) Supervisory control and data acquisition (SCADA) systems - both capable of recording vibrations. As opposed to standalone CM-systems, SCADA-data extracts data already being allocated at the wind turbine controller. Recordings are typically made every 10 second and averaged over 10 minutes (Verma, 2012). Due to the long sample intervals and the 10 minute averaging, the characteristic vibration signature gets corrupted and can not be used to detect concrete faults accurately. The successful utilization of SCADA-data has rather been recognized by its use in data-mining applications for fault diagnosis and prognosis (e.g, see Kusiak and Verma, 2012; Verma, 2012; Yang et al., 2013).

The dedicated equipment is characterized by high sampling frequencies. Thus vibration analyses using CM-data tend to be concentrated around detection of characteristic vibration signatures associated to the faults (Nie and Wang, 2013).

Another important piece of information is the fault logs. These allow for matching the signal characteristics or data-patterns, either it is from SCADA or dedicated CM-sensors, to the system status.

1.3 Turbine Data Not Easily Accessible

In his article on lack of data sharing in the renewable-energy industry, Kusiak (2016) describes long lasting back-and-forth communication across different energy companies including the signing of several non-disclosure documents to obtain the required data for analyzing wind turbine sensor data within his Iowa University community. Getting higher frequency data is even harder as it might require the permission from the sensor manufacturers. Also the turbine manufacturers plays an important role. As part of the warranty, the manufacturer typically provide maintenance and all related condition monitoring during the first five years (Salomonsen, 2015; Togstad, 2016). For competitive reasons, typically only aggregated values are made available for the operator during this period (Lund, 2016).

From the operator's point of view, where large parts of the wind farm project is relying on regulations and banking, the cost-efficient operation makes a crucial arena for obtaining their competitive edge (Togstad, 2016). In the bigger picture, the lack of data-sharing disables researchers and other knowledge-communities from enabling the full potential associated with better data-utilization.

1.4 Objectives

Statkraft, being the operator of several wind farms both onshore and offshore, has granted this project with access to vibration data from the CM-system of a land-based wind-farm. As neither SCADA-data or fault logs have been obtained, the following research-question becomes relevant:

RQ1: To what extent is it possible to monitor the condition of wind turbine components without the support from fault logs?

The main objective of the thesis is to develop develop and apply an approach for condition monitoring for the wind turbine gearbox. The main objective are underpinned by the following partial objectives:

PO1: Segregate the part of the vibration caused by load from those being attributed to degradation.

PO2: Relate the vibration-induced response to time.

To deal with the missing information the following key hypothesis is stated:

Hypothesis I Most of the data correspond to normal turbine behavior.

Other assumption dictating the approach are:

Hypothesis II All turbines can be considered equal, i.e., they give rise to the same vibration response under equivalent conditions.

Hypothesis III The vibration response can be modeled as a function of operational load and degradation level in an additive manner.

The true nature of these assumption are highly relevant for being able to choose an appropriate approach for meeting the main objective. Efforts to evaluate their validity are therefore continuously sought during the vibration analysis part of the thesis.

1.5 Limitations

Without fault logs there is no way to relate the observed vibrations to the system state experienced by the wind turbine. This unsupervised characteristic makes the answers to the research questions become reasoned arguments rather than definite answers.

1.6 Structure of the Thesis

The remaining report is organized as follows:

Chapter 2 - Theoretical Background

Fundamental maintenance concepts are explained along with an introduction to wind turbine functioning and vibration analysis.

Chapter 3 - Description of Acquired Vibration Data And Case-Study Outline

The vibration data that was available through the wind farm site server is described before parts of the data is selected to be the basis for the vibration analysis. Lastly, a component is selected and an outline for the analysis is given.

Chapter 4 - Methodology and Model Construction

The methodology is first explained through a simplified use-case. Next, the choice of enabling tools/techniques are justified before being formally presented. Ultimately, they are applied to the real turbine data.

Chapter 5 - Results and Discussion

Key findings from the degradation analysis are presented and discussed along with suggested further work.

Chapter 6 - Conclusions

The final accomplishments are summarized.

Chapter 2

Theoretical Background

This chapter introduce essential maintenance concepts. Most emphasis is put on condition based maintenance (CBM) due to its relevance and potential for cost-cuttings in the wind industry. The enabling building blocks for carrying out CBM is described through the prognostics and health management (PHM) cycle.

2.1 Maintenance Concepts and Vocabulary

The actions referred to in the definition on maintenance " ... the combination of all technical, administrative and managerial actions during the life cycle of an item intended to retain it in, or restore it to, a state in which it can perform the required function (Marquez, 2007)" are commonly described on three distinct but interacting levels (e.g, see Marquez, 2007; Ramírez, 2013).

- Strategic level (long-term): These are long time horizon decisions reflecting the business strategy and corporate visions by means of maintenance priorities. Such decisions typically result from commercial and economic considerations and may involve design, choice of enabling technologies, acquisition of requisite skills, etc.

- Tactical level (medium-term): Actions at the tactical level would determine the correct assignment of maintenance resources (skills, materials, equipment etc.) that enables fulfillment of the strategic maintenance priorities. Outputs from the tactical level is typically

detailed maintenance scheduling linking the type of maintenance resource to time and place.

- Operational level (short-term): While the two previous levels are more concerned with designing the maintenance framework, the operational decision-making would ensure an efficient implementation of the maintenance program. This involves e.g. on-time execution of work carried out by skilled technicians following the correct procedure. Moreover, an important function taking place at the operational level is the collection of relevant failure and maintenance data going into the maintenance information systems.

The interaction between decision levels is furthermore emphasized in figure in 2.1 showing the maintenance management loop. Its purpose is the enabling of continuous improvement that exploits accumulated knowledge and experience.

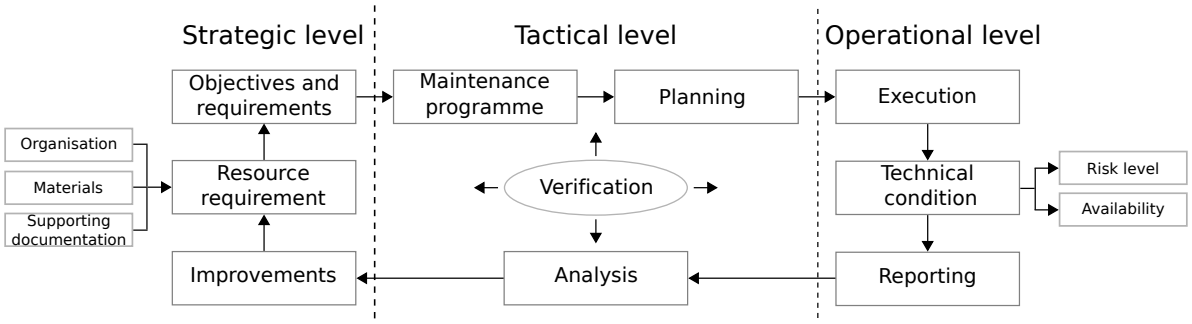


Figure 2.1: Maintenance management loop. Adapted from Oljedirektoratet (1998)

The definition on maintenance furthermore speaks about 'restoration' and 'retention' which leads to the discrimination of corrective and preventive maintenance policies respectively. Commonplace maintenance strategies under corrective and and preventive maintenance are shown in figure 2.2. The subsequent branching shows associated criteria for when to initiate maintenance followed by possible types of maintenance interactions. Although not listed in the figure, is should be noted that together with CBM, *opportunity maintenance* is particularly relevant to the offshore wind farm. Opportunity maintenance is characterized by seizing the opportunity of effective resource deployment - for example during periods where accessibility is beneficial, or by grouping maintenance actions while already on site.

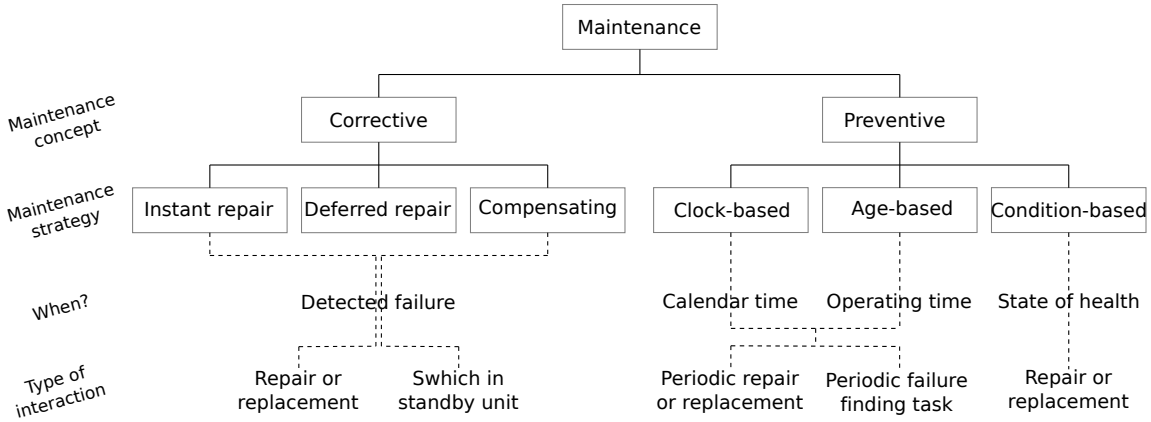


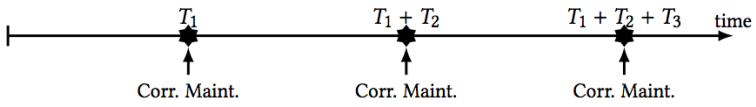
Figure 2.2: Classification of maintenance types. Adapted from Rausand and Høyland (2004)

A fundamental starting point for a discussion on maintenance policies would be the fact that we do not know when a component or subsystem will break down. In other words, the time to failure, T, must be considered a random variable.

☞ A **failure** is the permanent inability of a component or a system to perform its functions (Isermann, 2011).

The maintenance policy should then be selected based on how well T can be approximated and from what consequences that can be associated with a possible failure. Letting $\overline{C}$ represent the average cost of maintenance, and assuming that the component life-time distributions, f_T are available, the economic viability of the different strategies can be evaluated using system reliability theory. Under the regime of corrective maintenance - commonly referred to as run to failure (RTF) - *repair* and *corrective maintenance* are used interchangeably. Repair might be instant or deferred. As wind turbines are designed without redundancy, switching to standby is not an option.

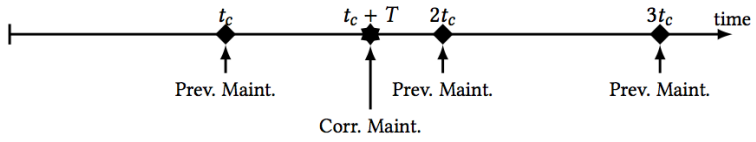
Using system reliability theory (further elaborated in Rausand and Høyland, 2004, Chapter 9), the average cost of corrective maintenance with immediate repair, $\overline{C}$, is shown in figure 2.3. Note that for the average cost $\overline{C}$ to be validly expressed, the repair make must be perfect, i.e., the component is restored to a state where it is considered *as good as new.*



$$\overline{C} = \frac{c+k}{\int_0^{+\infty} t f_T(t)\mathrm{d}t} = \frac{c+k}{\mathrm{MTTF}}$$

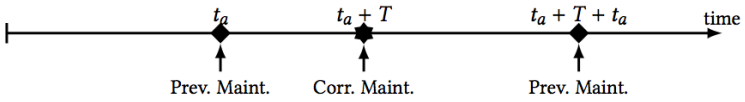
Figure 2.3: Operational timeline when running to failure with average associated cost per unit of time: $\overline{C}$. Figure from Lefebvre (2015).

A *clock-based* maintenance strategy maintains the items at regular time intervals ($t_c, 2t_c$ ...) regardless of operational age, which is the dictating parameter in scheduling *age-based* maintenance. As indicated in figure 2.4, both policies execute corrective maintenance (repair) also upon failure, meaning that the intervals should be carefully estimated.



$$\overline{C}(t_c) \approx \frac{c + (c+k)\int_0^{t_c} f_T(t)\mathrm{d}t}{t_c}$$

(a) Clock-based maintenance and average associated cost per unit of time: $\overline{C}$.



$$\overline{C}(t_a) = \frac{c + k\int_0^{t_a} f_T(t)\mathrm{d}t}{\int_0^{t_a} t f_T(t)\mathrm{d}t + t_a \int_{t_a}^{+\infty} f_T(t)\mathrm{d}t}$$

(b) Age-based maintenance and average associated cost per unit of time: $\overline{C}$

Figure 2.4: Preventive maintenance policies with predetermined parameters for a) calendar time, t_c and b) operational age, t_a. Figure from Lefebvre (2015).

As phrased in Rausand and Høyland (2004), "... condition-based maintenance is a maintenance policy where the maintenance action is decided based on measurement of one, or more, variables that are correlated to a degradation, or loss of performance, of the system". If a CM system were to provide perfect prognosis, it would enable for *ideal* maintenance as demonstrated in figure 2.5. Maintenance is then carried out just before the component fails, meaning that its useful life is fully exploited. The need for corrective interventions is thus eliminated. However, in practice, ideal maintenance should be left a theoretical curiosity.

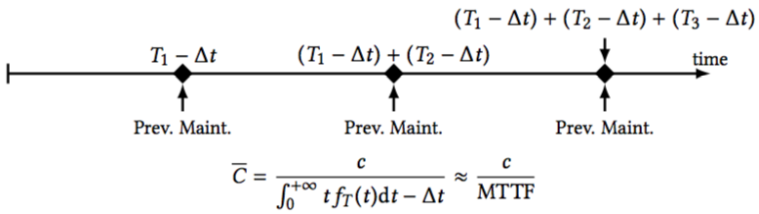


Figure 2.5: Operational timeline with ideal maintenance and the average associated cost per unit of time: $\overline{C}$. Figure from Lefebvre (2015).

In reality uncertainty gets introduced at several stages. For one, the prognostic analysis in it self is inherently uncertain; for two, the resource availability might not always be in the full control of the maintenance organization. In addition there might be wild-cards such as weather dictating the feasibility for timely maintenance. To illustrate how uncertainty influence the average cost, a small numerical application was simulated in MATLAB. Letting $c = 50$ and $k = 50$ represent the cost of corrective and preventive maintenance respectively and letting the component lifetime be normally distributed according to $T \sim N(375, 50)$, the average resulting costs are shown in figure 2.6.

The horizontal axis is relevant only for the condition-based maintenance policy. ΔT represent when, relative to the prognostic output, the maintenance action is carried out. For good prediction accuracy, i.e., for smaller standard deviations in the prognostic output, it is desirable to make the maintenance action shortly before the predicted failure date. For the same reason, the penalty cost increase rapidly as soon as $\Delta T = 0$ is exceeded. When the prognostic uncertainty increase, i.e., for larger standard deviations, more margin needs to be taken into account. The plots furthermore highlight that fixed-interval strategies might very well outperform CBM when based upon unreliable predictive information. Associated MATLAB code is given by `PredAccuracy.m` in appendix B.1.1.

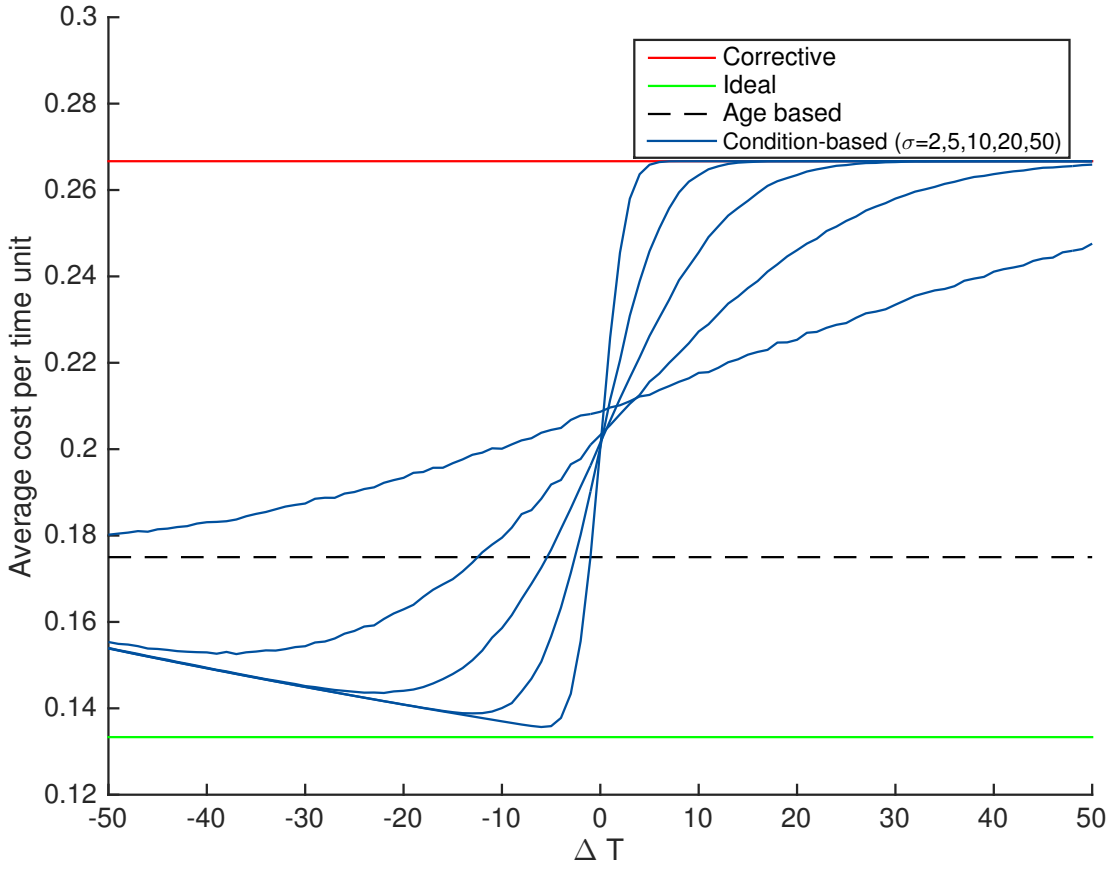


Figure 2.6: Relations between maintenance strategy and cost. For condition-based maintenance, the figure shows the average cost per time unit for different prognostic performance levels as a function of *when* the interaction happens relative to the predicted failure date. It is shown that a certain level of predictive accuracy is required to outperform a time-based policy with reasonable intervals.

It is stressed that CBM need not to rely on early warning prognostic information, but might be based on deviating observations obtained from real-time monitoring calling for more or less immediate actions. This could for instance involve characteristic vibration signatures that triggers immediate actions such as shutdown or inspection. This might however in many cases be too late, which calls for prognostic information supporting the CBM-regime with early notice of the incipient faults. How such information is obtained can best be explained through the prognostic and health management cycle.

2.2 The Prognostics and Health Management Cycle

Vachtsevanos et al. (2006) expresses the purpose of prognostics and health management (PHM) as "... to, from sensor data, detect and isolate incipient faults and as quickly and accurately as possible envisage a fault to failure progression timeline anticipating when a failure will occur." The PHM-cycle can be described using the building blocks in figure 2.7.

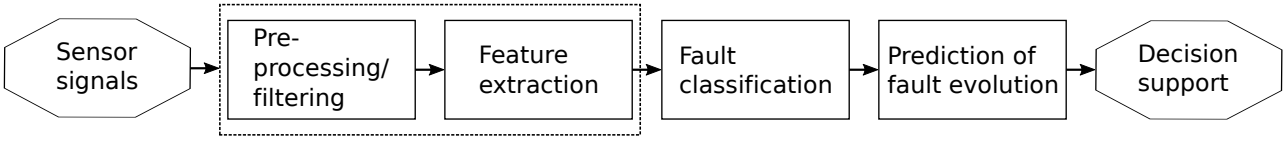


Figure 2.7: Required building blocks for PHM. Adapted from (Vachtsevanos et al., 2006).

2.2.1 Sensor technology

Sensors constitute the very basis for fault diagnosis and prognosis. The sensor in it self serves only as a device capable of sensing change in some physical quantity (Collacott, 1977, chapter 3). Next, the energy in the measured signal is received by a transducer whose task is to transmit the data, often as a digitized electrical signal to some data-base for processing (Vachtsevanos et al., 2006, Section 3.2.1).

2.2.2 Data processing

The data processing can be considered a two-step process, where the *pre-processing* intends to enhance the signal characteristics to facilitate for efficient extraction of information, that is, the indicators of the condition of a failing component. This involves filtering, amplification, data compression, data validation and denoising - all techniques intended remove noise and/or signal characteristics that are otherwise misleading (Vachtsevanos et al., 2006, section 4.3.1). *Feature extraction* is the process of turning data into information, i.e., the mapping from raw signals to meaningful signatures. The objective is to select appropriate features that are able to indicate the state of health for the component or subsystem (Vachtsevanos et al., 2006, Section 4.4). For rolling impacts, e.g., coming from a bearing, the energy present in the emitted frequency band is a well-proven feature.

2.2.3 Fault classification

To detect a fault, it is satisfactory to know the normal behavior of the monitored system. This follows from the below definition of a fault:

☞ A **fault** is an abnormal condition that may cause a reduction in, or loss of, the capability of a functional unit to perform a required function (IEC61508).

If a point is registered outside the normal behavior area, there is a fault. Referring to figure 2.8, the gray point represents a *fault detection* as the system behavior is inconsistent with the normal behavior: $(U,Y) \notin \beta_n$. The red point close to the center of β_{f1} represents both a fault detection and a *fault isolation*. The fault is said to be isolated because the point can be associated with one specific faulty behavior: $(U,Y) \in \beta_{fi}, \forall i = 1,...,n$. For the red point contained in the overlap between β_{f1} and β_{f2} there is no fault isolation, as the point represents more than one faulty behavior. Because the green point is consistent with both the normal behavior and faulty behavior 1, neither a fault detection or a fault isolation can take place (Blanke et al., 2006).

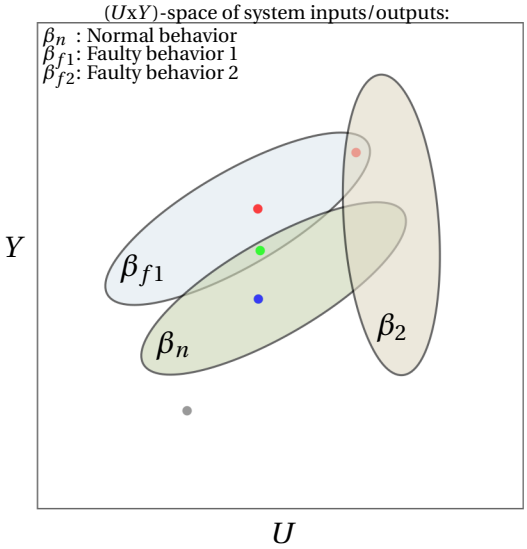


Figure 2.8: Data points representing system input/output associated with different types of known and unknown system behavior.

The field of fault detection and isolation is now a mature science allowing for well-proven techniques both from the domain of physical and data-driven models (Nie and Wang, 2013; Ata, 2015). Physical modeling rely on an accurate dynamic model constructed by physical model

parameters representing the system in question. The data-driven approach does on the other hand employ a model that is trained from data. By comparing the input/output from the actual system with the model response, the resulting discrepancy or *residual* is used extract information regarding current system health. The flow of this exercise is depicted in figure 2.9.

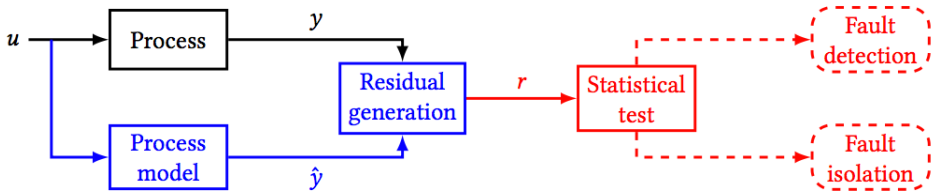


Figure 2.9: Process flow showing how deploying a model allows for calculating residuals for fault detection/isolation purposes. Figure from Lefebvre (2015)

Based upon several authors (eg, see Aldrich and Auret, 2013; Verma, 2012) these reoccurring advantages can be linked to the data-driven approach.

- Cheap exploitation of the ever-growing volumes of process data accumulating from technical systems.
- Uses few or no assumptions - do not require accurate dynamic models of the physical system under study.
- Unfortunately, complete knowledge of real processes is often not available or very expensive to acquire, thus complicating the construction of an accurate physical model.

2.2.4 Prognostics

Prognosis is a composite word consiting of the Greek words pro and gnosis, which literally translates into the ability to acquire knowledge (gnosis) before (pro) a future event will occur. This is not a new idea; two thousand years ago there were oracles scattered all over Greece and Italy and some, such as Delphi and Trophonios, grew into large and wealthy organization (Stopford, 2009). In the PHM context, the prognosis is understood as the precise and accurate estimation of the remaining useful life (RUL). Although large progress has been achieved in PHM community recent years, estimating the RUL still remains an the Achilles' heel (Vachtsevanos et al.,

2006, Section 6.1). This is true due to several reasons. Above all, providing reliable RUL estimates places great demands on the available data (on top of what is required for doing the fault-detection/classification step). Firstly, you would need the duration between the point in time where the incipient fault(s) occurs and the point in time when the failure occurs. Secondly, you would need access a adequately detailed data telling what load the system has been exposed to. This is particularly true for the wind turbine. A fault to failure progression time-line resulting from harsh weathering and excessive loads to the system might not be relevant for situations where the load is moderate. Therefore, weather forecasting models should be employed to substantiate for an appropriately chosen RUL-estimate for wind turbines. Ultimately, you would need statistically sufficient samples - often meaning very large data-sets due to the large-grain uncertainty entailing the field of prognostic analysis.

2.3 Wind Turbine Fundamentals

2.3.1 Functional breakdown

Put simply, a wind turbine works the opposite way of a fan. Instead of consuming electricity to make wind, the wind is used to make electricity. The energy conversion process including the voltage step-up transformation required for efficient transportation of electricity is covered by the first four primary functions (from the left) in figure 2.10. Remaining primary functions can be seen as facilitators: Cooling prevents the system from catching fire and the structural support & housing provides i) a reasonable position for the blades; and ii) a closed environment shielding the components from ambient impacts. The secondary functions show in further detail how their parent functions are achieved together with associated components. For a complete introduction to the wind turbine functioning, see e.g. Hau (2006, 2013).

Within the P6 auxiliary functions, the wind farm maintenance regime serves as a required function allowing for an acceptable up-time. This is indicated with by the red dashed box and would involve maintaining each branching going out from the system function. The complete condition monitoring system for a modern turbine is composed of many sensors, each covering particular sub-assemblies of the wind turbine. Among the variety of techniques for wind turbine condition monitoring, the following will be restricted to encompass vibration analysis. Readers are referred to Rolfseng (2015) for a more complete presentation capturing the variety of wind turbine CM techniques.

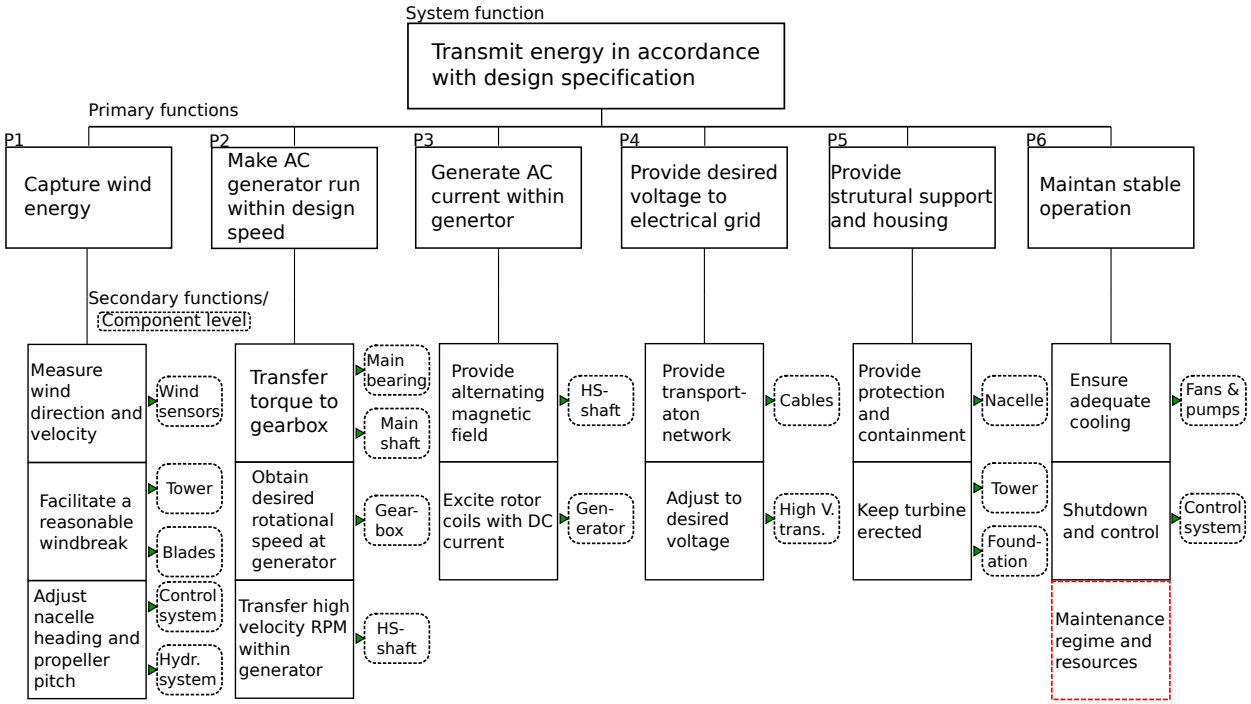


Figure 2.10: Functional breakdown structure for a wind turbine with componentes linked to final functional indenture level and associated components. The breakdown structure is defined by the author, but relies on wind turbine theory presented in Hau (2006, 2013).

2.3.2 Vibration analysis

Vibration analysis (VA) is the most well-known CM techniques for rotating equipment (Cornelius, 2004). For wind turbines VA is predominantly used for shafts, bearings, gearboxes, blades and generators (Tchakoua et al., 2014). This set of components are commonly referred to as the *mechanical drive train* (Hau, 2006).

The standard VA sensor configuration for the Vestas V117 has 13 sensors as shown in figure 2.11. Figure 2.13 shows that among different ways of detecting faults in rotating equipment, VA typically provides the earliest fault detection. How early, and at which accuracy is largely dependent on the rotational speed of the equipment (Salomonsen, 2016). For faster rotating parts, the energy given from the rolling impact of a small damage is much larger and therefore more clearly visible from the vibration readings. Slowly rotating parts include the main shaft and bearing, the blade bearing as well as the the first stages of the gearbox. Remaining gears, bearings and shafts are then fast rotating. For this reason one can expect a better hit rate and prediction time for damages in the fast rotating bearings and gear components. It is also a fact that the high speed rotating components have a higher wear rate and thus results in a generally higher failure rate than the slowly rotating parts of the drivetrain (Salomonsen, 2016).

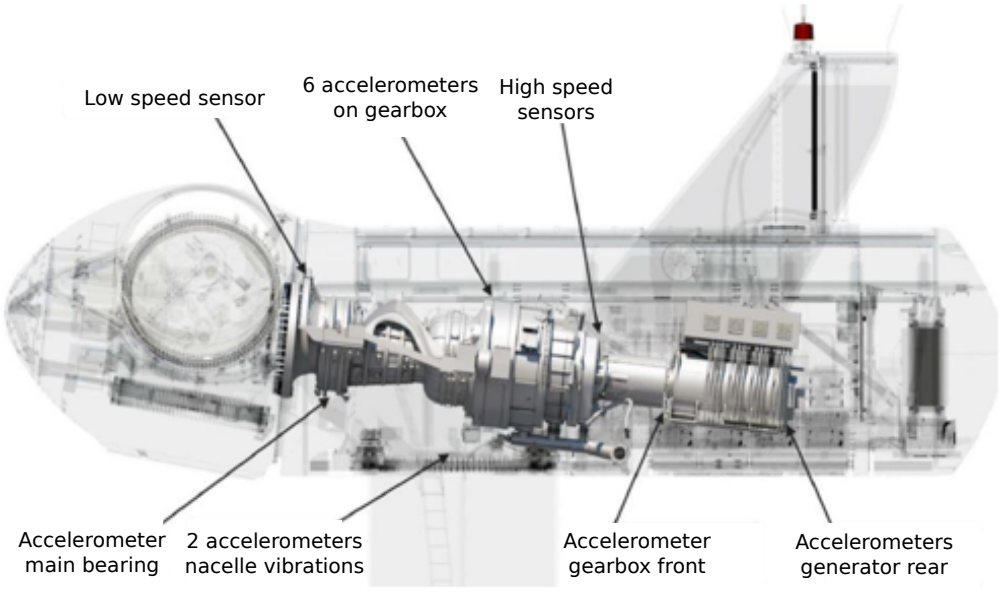


Figure 2.11: Main components shown together with standard sensor configurations in a Vestas V117 3,45 MW wind turbine. Figure from Salomonsen (2016).

The vibration measurements are collected through *accelerometers*. When the base on which the accelerometer is fitted start to vibrate, it exerts a force onto the accelerometer. This force is captured by piezoelectric discs inside the sensor which generate an electrical signal that is directly proportional to the accelerations (Vachtsevanos et al., 2006). The process of obtaining accelerometric data as digital read-outs is shown in figure 2.12.

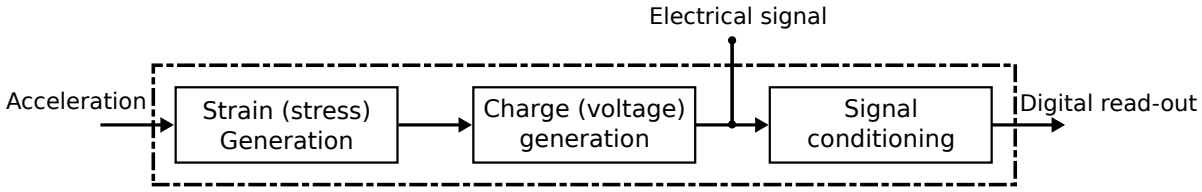


Figure 2.12: Operation of a piezoelectric accelerometer. Based upon Vachtsevanos et al. (2006) and Collacott (1977).

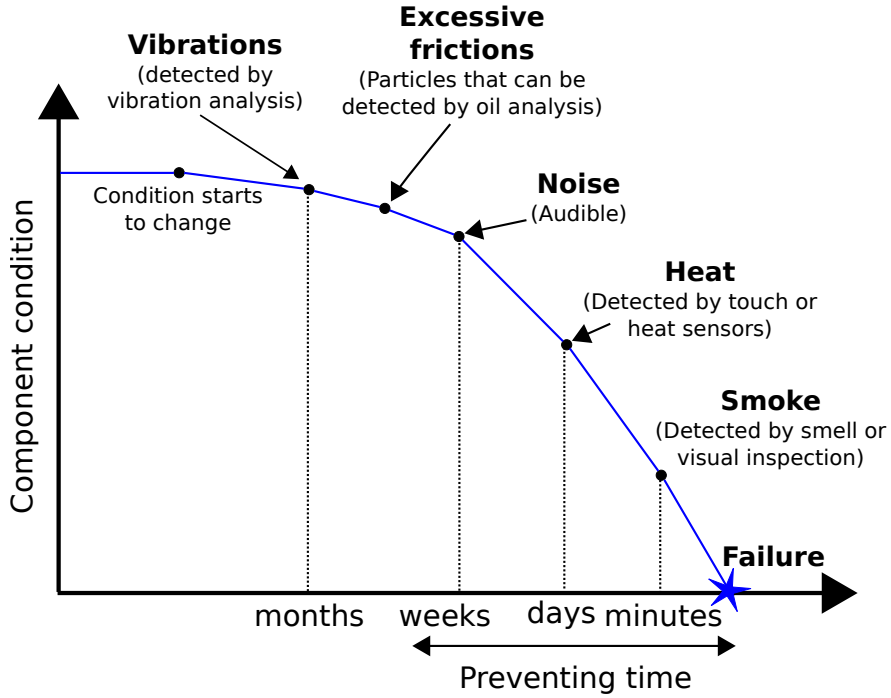


Figure 2.13: Vibration analysis can provide relatively early warning when compared to alternative techniques for anomaly detection. Adapted from Madsen (2011).

A well-proven technique for vibration analysis is the Fast Fourier Transform that enables for frequency domain representation of the vibration (Cornelius, 2004). The interest of the frequency-domain representation becomes clear when considering the complex wave composed of multiple sinusoidals. The fourth row of figure 2.14 shows the sum of the three above

time-series, each of which when considered alone can be fairly easily interpreted. The problem arise when they are added together, such as would be the case for most rotating machines.

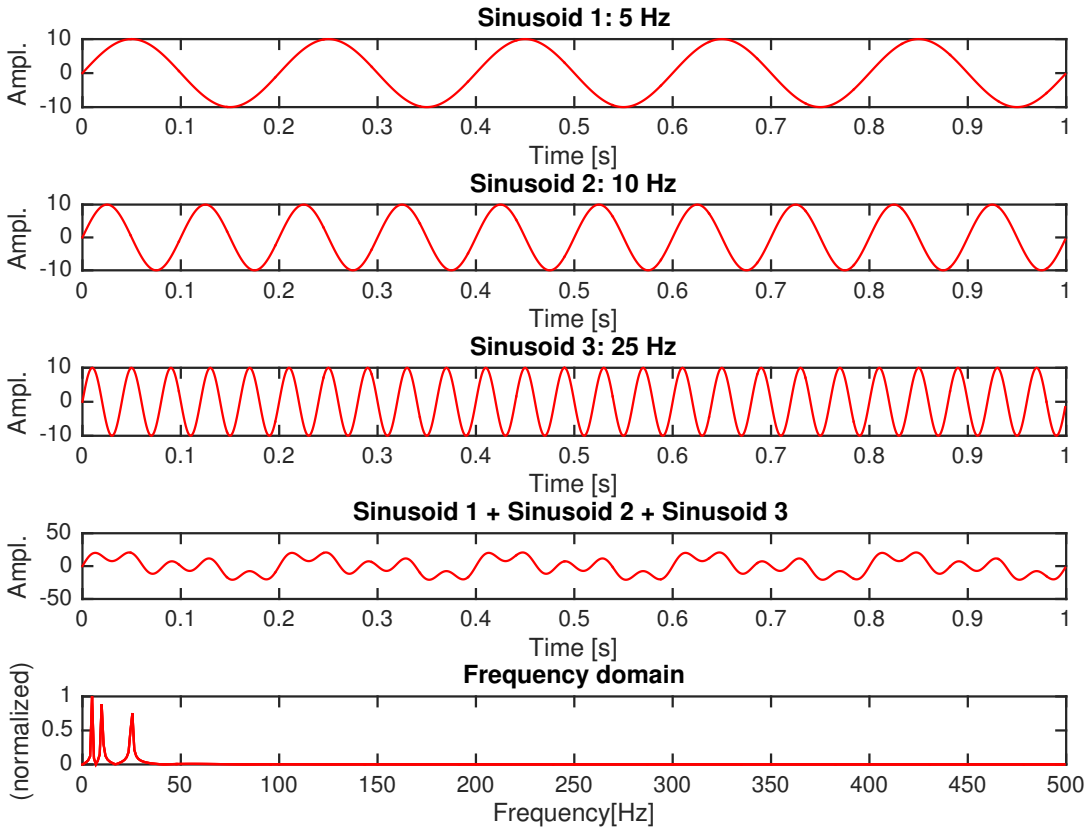


Figure 2.14: Figure shows three distinct sinusoidals separately before being added into a complex composite waveform. Lower plot shows the resulting frequency domain representation. Associated MATLAB-code found in appendix B.1.2.

To decompose such complex wave-forms, the FFT-algorithm has proven superior in obtaining a computational efficient approach for mapping the time-series into the frequency-space (Redmon, 2002). When applied to the complex wave form, the frequency domain representation in the bottom plot shows clearly what contributions that together formed the complex wave.

What makes the FFT so useful is the fact that many faults can be recognized from their characteristic frequency signature (Jardine et al., 2006). However, for variable load systems, such as the wind turbine, particular attention must be given. As wind in nature is irregular and drives the

wind turbine unsteadily, there is a need for separating the part of the vibration signal that is attributed to degradation from the part simply being a result form operational load. To still be able to obtain frequency-domain vibration signatures that are characteristic of faulted states, a variety of techniques are suggested (e.g., see Chaari et al., 2012; Antoniadou et al., 2015). Whereas these are techniques intended for real-time visual inspection of the frequency characteristic, they might not be efficient in terms of monitoring the *evolving* fault (Samanta, 2004). Monitoring of the evolving fault would typically entail large amounts of historical data that covers the entire range - from the incipient fault until the event of failure.

Chapter 3

Description of Acquired Vibration Data And Case-Study Outline

The vibration data was obtained March 17^{th} during a visit at Statkraft's headquarters at Lysaker, Oslo. Statkraft is a leading company in hydropower on an international scale and Europe's largest generator of renewable energy. Their significant commitment to wind energy is manifested through ownership and operation of land-based wind farms in Norway, Sweden and the UK as well co-ownership to the Sheringham Shoal wind farm off the coast of North Norfolk[1]. The data for this project is accelerometric data acquired from a land-based wind farm.

3.1 Wind Turbine Data Management And Structure

Per date the monitoring of the wind farm is carried out by the vendor. Statkraft themselves have a *passive connection* to the monitoring data, essentially meaning that parts of the data is made available for them. To manage what data to be pulled from the turbines, an active connection is required, that is, the type of connection possessed by the vendor. Parts of the condition monitoring services, such as the hardware and server-setup, are externally provided by a company having materialized their expertise through condition monitoring services.

The backbone of the CM-system is the data acquisition and analyzing unit. Being placed inside the nacelle, the unit synchronously collects and analyze data from a number of connected

[1] http://www.statkraft.no/Energikilder/Vindkraft/

accelerometers before the information is sent to the wind farm site-server and/or controller. Furthermore, a number of counters (e.g. RPM, oil, anemometers) can be coupled to the unit, allowing for more meaningful information depending on what type of analysis is carried out. Through an internet-based user-interface users can request data for data mining and analysis purposes.

With regards to data-extraction, the available options are shown in table 3.1. As will become apparent, the data acquisition and analyzing unit has the computational capacity of readily presenting a set of features for the end user to download - not only raw signals in time-series format.

Table 3.1: Site server user-interface options

Location(s)	Which turbine(s) to consider.
Sensor	What component to consider
Measurement	What feature to consider
Condition	Records associated to a given power-range
From / to (yyyy-mm-dd)	Define time horizon
Number of records	Define max. number of records to extract
Output type	File format

Along with the actual feature measurements, each record are stored with the following auxiliary information:

Table 3.2: Table showing what information is stored in a record and how it is structured.

Turbine number	Rec. number	Power-bin (lower)	Power-bin (upper)	Power (avg.)	Yaw-angle	Rotor RPM	Wind speed	Generator RPM	Date/ time

The below lists shows the available sensors (components) and conditions (power bins).

Components

- Generator
- Generator (rear)
- Highspeed shaft
- Main bearing
- Intermediate shaft (IMS)
- Planetary stage of gearbox (Planet)

Power bins

- 0-920 kWh
- 921-1150 kWh
- 1151-1403 kWh
- 1404-1656 kWh
- 1657-1909 kWh
- 1910-2185 kWh
- 2186-2415 kWh

As there is 6 component and 7 power bins, there exist 42 combinations for every feature. It was a total of 40 available features, which makes the total number of combinations 1680 - all potentially containing vast amounts of records. Due to the limited time during our visit, it was not possible to obtain all combinations. Hence, some prioritizing was made.

3.2 The Collected Data

3.2.1 Feature "Time _8000"

Initially, selecting all turbines, we tried to collect the feature Time _8000 for all sensors and all power bins (42 combinations). 8000 denotes the maximum value of the frequency spectrum from 0 to 8.138 kHz. For every record there is 32000 samples collected over 1.5360 seconds, yielding a sampling frequency, f_s , of 20833 Hz.

Having the complete time series would be ideal because all other features can be derived from it. However, it turned out that only the highest power-bin contained data. The Resulting data gathered from the time series is displayed in table 3.3.

Table 3.3: Data capture for Time_8000 (time series with frequency spectrum ranging from 0 to 8 kHz).

	Feature considered: Time_8000		
Component	Power output bin	No. of turbines screened for data	No. of turbines that provided data
Generator	2186-2415	68	48
Generator (rear)	2186-2415	68	48
Highspeed	2186-2415	68	48
Intermediate shaft (IMS)	2186-2415	68	48
Planetary stage of gear (planet)	2186-2415	68	48

The 48 turbines that contained data were all commissioned in a second phase of the wind farm project, i.e., they were all put to operation at the same time. The fact that the other turbines did not contain data could be due to a different type of monitoring regime shared among these turbines. Figure 3.1 two shows the distribution of records for each of the components per year.

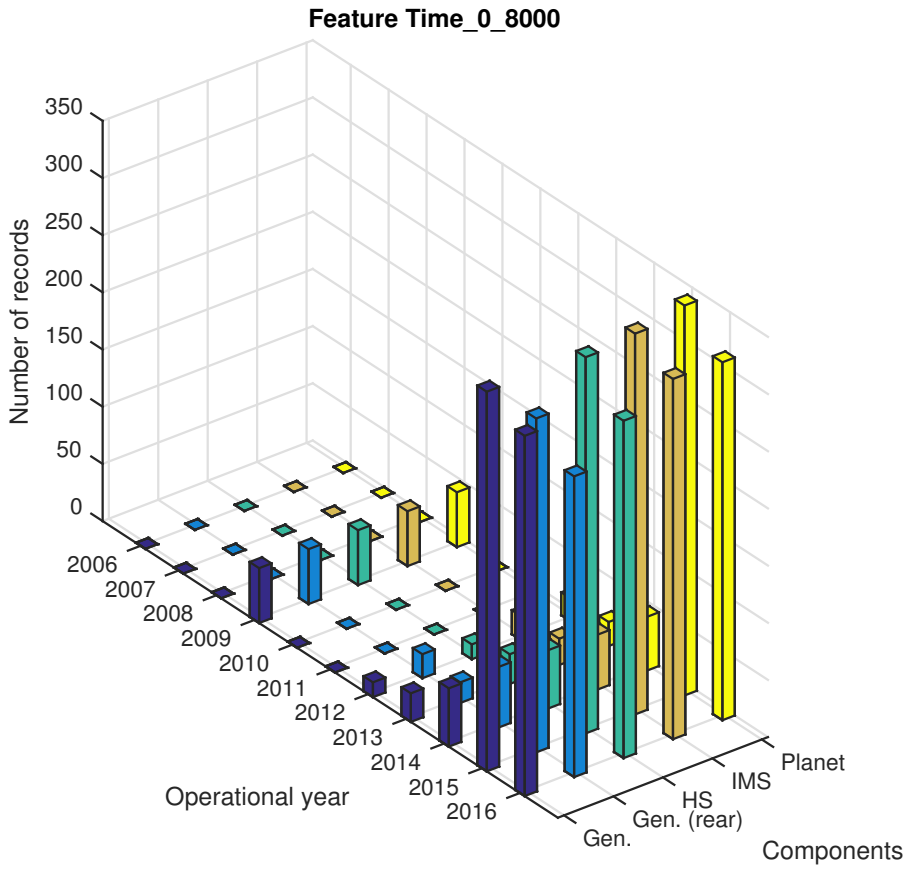


Figure 3.1: Annual number of records for each component.

3.2.2 Feature "FFT_0_8000"

The feature FFT_0_8000 was successfully acquired for all sensors and all bins. Again 8000 denotes the frequency spectrum ranging from 0 Hz to 8138 Hz. For every record, the number of samples is 401. The associated resolution, Δf , is hence 20,34 Hz. The resulting data capture from the feature FFT_0_8000 is shown in table 3.4. The number of turbines screened for data was reduced from 68 to 20 after finishing the generator due to the limited time available. Considered turbines are T21-T40, all members of the second commissioning phase.

Table 3.4: Data capture for FFT_0_8000 (frequency domain vibrations in the range from 0 Hz to 8.138 kHz.)

Feature considered: FFT_0_8000			
Component	Power output (bin)	No. of turbines screened for data	No. of turbines that provided data
Generator	0-920	68	48
Generator	921-1150	68	48
Generator	1151-1403	68	48
Generator	1404-1656	68	48
Generator	1657-1909	68	48
Generator	1910-2185	68	48
Generator	2186-2415	68	48
Generator (rear)	0-920	20	20
Generator (rear)	921-1150	20	20
Generator (rear)	1151-1403	20	20
Generator (rear)	1404-1656	20	20
Generator (rear)	1657-1909	20	20
Generator (rear)	1910-2185	20	20
Generator (rear)	2186-2415	20	20
Highspeed shaft	0-920	20	20
Highspeed shaft	921-1150	20	20
Highspeed shaft	1151-1403	20	20
Highspeed shaft	1404-1656	20	20
Highspeed shaft	1657-1909	20	20
Highspeed shaft	1910-2185	20	20
Highspeed shaft	2186-2415	20	20
IMS	0-920	20	20
IMS	921-1150	20	20
IMS	1151-1403	20	20
IMS	1404-1656	20	20
IMS	1657-1909	20	20
IMS	1910-2185	20	20
IMS	2186-2415	20	20
Planet	0-920	20	20
Planet	921-1150	20	20
Planet	1151-1403	20	20
Planet	1404-1656	20	20
Planet	1657-1909	20	20
Planet	1910-2185	20	20
Planet	2186-2415	20	20

Figure 3.2 two shows the annual distribution of records for each of the components. Compared to Time_8000, the amount of data is substantially larger for this feature. Part of the explanations is the fact that all power bins contained data.

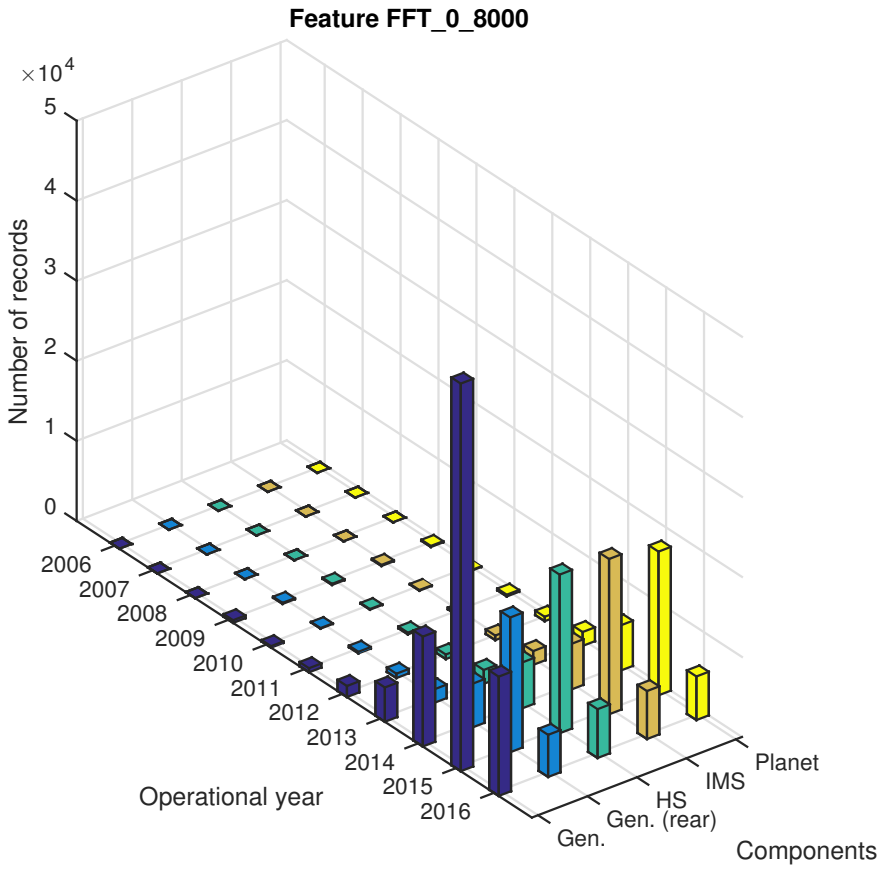


Figure 3.2: Annual number of records for each sensor.

3.2.3 Less structured data capture - a mix of features and components

The last set of features collected was initiated based on suggestions from Statkraft staffing who believed that these particular features had been subject to some previous analysis (without knowing the specifics). This included Env_100, Env_300 and FFT_127p, all of which are denotations of envelopes with different frequency spectra. At this point, the time available for extracting more data does not allow for being as systematic as previously. This becomes even more clear when realizing the vast amount of records associated with these features. To remedy

the substantial download-time, fewer turbines are selected for some of the features. Also, just the lower and upper power bins were requested. Resulting data captures are summarized in table 3.5.

Table 3.5: Features considered: Envelopes (frequency range 0-100 & 0-300) and FFT_127p .

Mix of features selected based on suggestions from Statkraft				
Component	Feature	Power output (bin)	No. of turbines screened for data	No. of turbines that provided data
Main bearing	Env 100	0-920	20	20
Main bearing	Env 100	2186-2415	5	5
Main bearing	Env 300	0-920	1	1
Main bearing	Env 300	2186-2415	20	20
Planet	FFT 127p	0-920	3	3
Planet	FFT 127p	2186-2415	20	20

Figure 3.3 shows the distribution of records for each of the components per year.

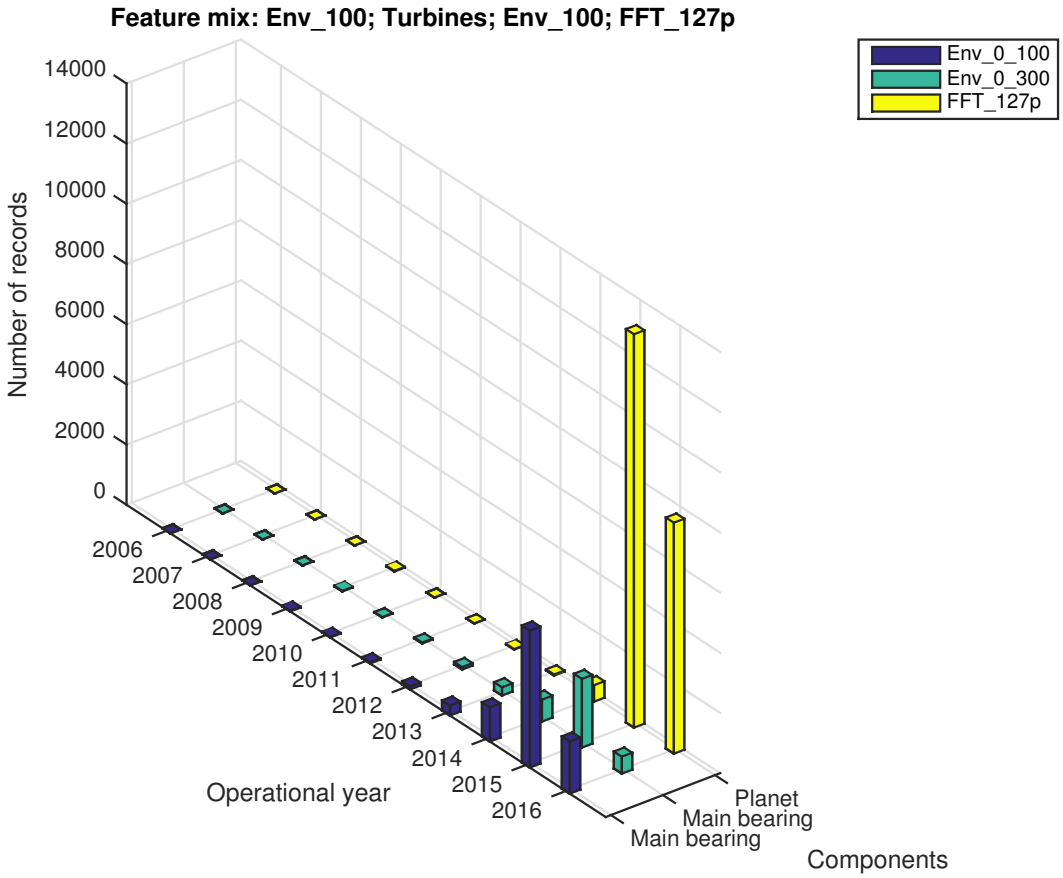


Figure 3.3: Annual number of records for each sensor.

3.3 Selecting Data and a Component for the Vibration Analysis

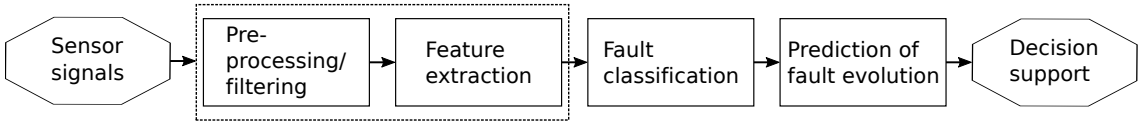
As mentioned, several combinations was empty. Why they were empty can only be revealed when knowing the data-pull-setting as well as the criteria for data-storage. Among the collected data, FFT_0_8000 yields the most complete dataset. For the operational years containing data, data was found in all power bins as well as for all sensors. Furthermore, the data capture for this feature is consistently based upon turbines commissioned at the same time. FFT_0_8000 is hence chosen for the further analysis.

A component which has received much attention is the wind turbine gearbox. For one, it is subject to varying load, calling for a way to segregate signals representing deterioration from those just being an effect of load. It also is a large and heavy component requiring laborious interventions to replace. Furthermore, by contributing with 13 % of the total sub-assembly cost (Hau, 2013) it is also by far the most expensive component. Faulstich et al. (2011), who extract reliability statistics for onshore wind turbines based on a comprehensive database, finds that the gearbox has the largest mean annual contribution to downtime and causes an average downtime of 18 days when subject to a major failure.

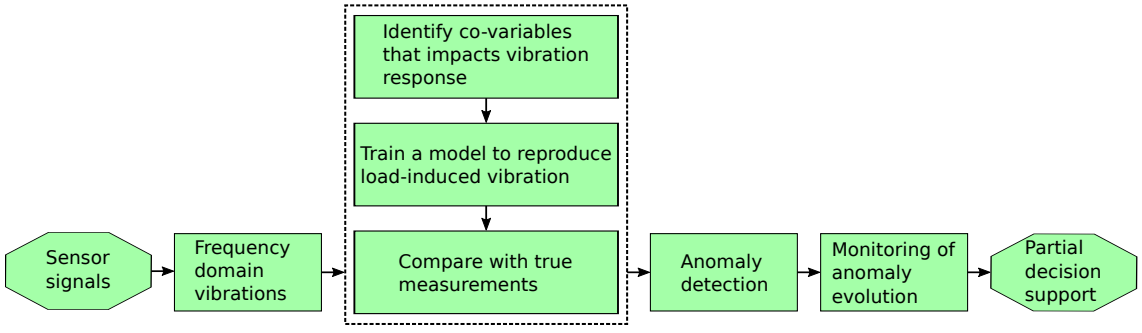
More specifically, the data is covering the planetary stage of the gearbox. The turbines under consideration, 2.3 MW machines, are equipped with a custom-built, three stage planetary-helical design. The high-torque stage, i.e., the slowly rotating side, has a planetary-helical design, whereas the intermediary and high-speed stages are normal helical stages (SIEMENS, 2009). In general, the fact that the records are made on the slowly rotating side makes the analysis more challenging, as the energy impacts are less than for the fast revolving case. Together, the abovementioned aspects made the gearbox appear as an interesting component to analyze, and is hence chosen for the further analysis.

3.4 Fitting the Analysis into the Framework of Prognostics and Health Management

Recalling from section 2.2 that the purpose of the prognostic and health management cycle is "... to from sensor data, detect and isolate incipient faults and as quickly and accurately as possible envisage an fault to failure progression timeline anticipating when a failure will occur." Without fault logs to confirm the system status, it is not possible to fully satisfy this objective. However, by means of outlier-detection and cluster-analysis it may be fully possible to reveal patterns in the data being representative of faulted states. Next, by monitoring the time-dependent presence in the assumed faulted states, the evolving anomalous behavior can readily be monitored. Thus, with some adaptations to the building blocks constituting the original PHM-loop, the PHM-framework is still relevant for the current analysis. How the analysis carried out in this project maps onto the original PHM-cycle is indicated in figure 3.4. 'Partial decision support' in the outputting end refers to the fact that when combining the findings with the missing information, valuable decision support could result.



(a) The PHM-cycle.



(b) Modified version of PHM-cycle applicable to the vibration analysis

Figure 3.4: Figure shows how the current analysis fits into the PHM-framework.

Chapter 4

Methodology and Model Construction

For clarity reasons this chapter starts off by describing the methodology in an overall manner. Key principles are communicated using a simplified and artificial use-case. This is followed by justifying the selections of tools/method enabling for the realization of the approach. After formally explaining their theoretical foundation, tools and techniques are applied to the actual turbine data.

4.1 Global Methodology Explained with Simplified Use-Case

The analysis seeks to enable condition monitoring for the planetary stage of the gearbox. To approach this task, a model generating vibration that corresponds to a given set of operational conditions will be developed. A key challenge in this respect will be to segregate vibration simply arising from higher operational load from those being a consequence of degradation. A crucial step in this respect is determining the set of co-variables that impacts the vibration response. For the case where the model is provided with *all* impacting co-variables and is able to give an accurate generalization, the residual will *exclusively* contain the contribution from degradation. Figure 4.1 expresses the approach schematically.

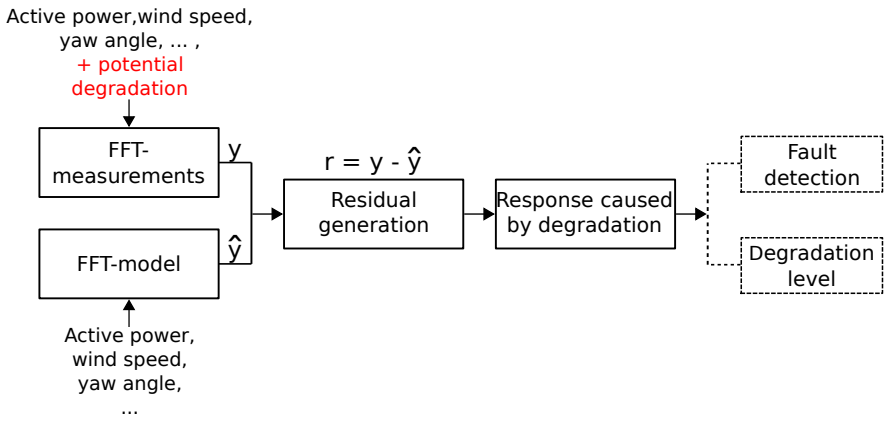


Figure 4.1: Condition monitoring approach.

Since there is 401 frequency increments in the spectra from 0 - 8138 Hz, the complete model will be composed of 401 sub-models - one for each frequency increment. Together, they will cover the complete frequency spectrum.

4.1.1 Key modeling assumptions and intended outcome

Due to lack of fault logs or other information that together with the FFT records could confirm the equipment status, we do not *know* what records are corresponding to normal behavior and not. To bypass this issue it is assumed that most of the records correspond to normal behavior, allowing for outlier detection to be a viable approach for fault detection. A second paramount assumption is that all turbines can be considered equivalent to each other and thus give rise to the same frequency responses. Considering the fact that all turbines are the same make, commissioned at the same time and share more or less the same coordinates speaks for this to be true. The validity of this assumption has implications for whether the entire wind farm can be considered at once or whether each turbine should be analyzed separately.

Hypothesis I : Most of the data correspond to normal behavior.

Hypothesis II : All turbines can be considered alike, i.e., they give rise to the same vibration response under equivalent conditions.

Hypothesis III : The vibration response can be modeled as a function of operational load and degradation level in an additive manner.

A topic that has been addressed by several authors is the challenge of segregating vibrations that simply results from load from those who are due to degraded items. By operating conditions it is meant active power, wind speed and/or other properties that influence the emitted frequency response. This can be expressed mathematically by letting vibration amplitude, operating condition and degradation be denoted y_f, op and d respectively:

$$y_f \sim f_1(oc) + g(d) \tag{4.1}$$

where oc and d are assumed to be additive contributions to the vibration amplitude:

$$y_f \sim f_1(oc) + g(d) \tag{4.2}$$

The decomposition in eq. 4.2 emphasize the fact that op and d are assumed to form the resulting vibration in an additive manner. What ultimately is sought is the part of the vibration responses that arise due to degradation, $g(d)$. To achieve this quantity, a model approximating $f_1(op)$ is developed

$$\hat{y_f} \sim \hat{f_1}(op) \tag{4.3}$$

which in turn allows for $g(d)$ to be approximated through the residual, Δf:

$$g(d) \sim \Delta f = \hat{f_1}(op) - f(op, d) \tag{4.4}$$

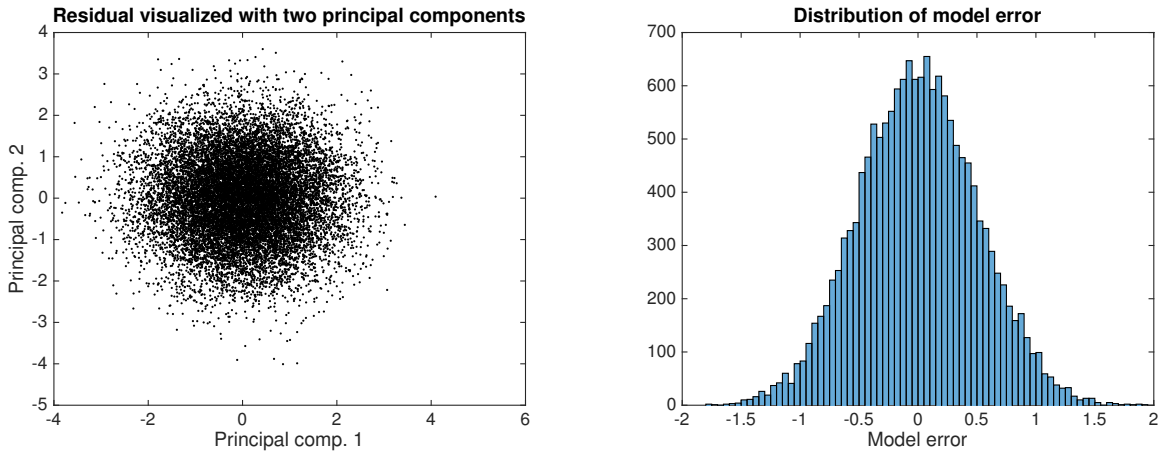
The model, being fed only with operating conditions, will not be able to reproduce the presumed additive contribution from degradation. Relying on the stated assumptions and the fact that the model is sufficiently well performing, the resulting residuals will then represent the contribution from degradation. Before being applied to the real turbine data, the approach is more easily communicated using a simplified use-case. The use-case is artificial, but emphasize key principles and how analyzing the residuals can provide valuable results.

4.1.2 Simplified use-case

The use-case assumes a *known* simple linear relation between vibration output $y(f)$ and input as shown in equation 4.5

$$y(f) = k + C_P \cdot P + C_Y \cdot C + C_{WS} \cdot WS + C_D \cdot D + \varepsilon \tag{4.5}$$

where C_P, C_Y and C_{WS} are coefficients for the rated power, yaw angle and wind speed respectively. k denotes a constant term whereas degradation and noise is represented by C_D and ε respectively. In other words, the complete set of co-variables for this demonstration comprise *rated power, yaw angle, wind speed* and *degradation*. For this simple demonstration, vibration amplitudes are generated only for two frequency increments. Since the vibration response is given through a linear relation, a simple linear regression model is used to estimate the vibration output. For the case where no degradation is present, the below plot shows the residual (left) and the distribution of model error (right).



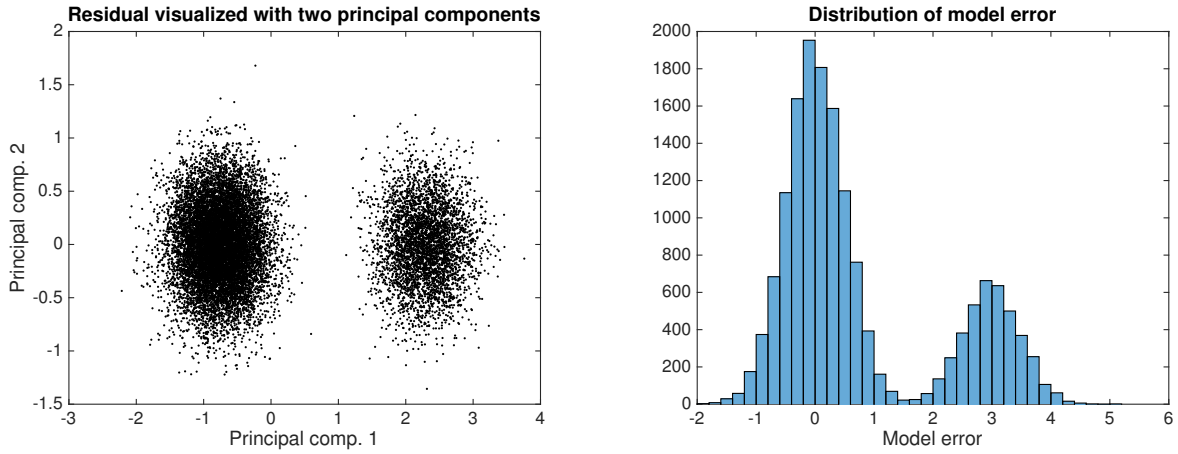
(a) Homogeneous cluster

(b) Normally distributed model error

Figure 4.2: Left plot shows a homogeneous residual with normally distributed errors (right).

The homogeneous cluster depicted above is as expected because the model have all required input to properly model the the vibration as there is no degradation. Due to the same reason, the model error is normally distributed. For the next experiment, some degradation is added by letting D assume non-zero values. As this parameter is not provided as model input, the model suffers from lack of information and will not be able to reproduce the part of the vibra-

tion response being caused from degradation. This piece of lacking information manifests itself through the smaller cluster in figure 4.3, representing the vibration responses being attributed to degradation. Whereas this figure shows a sudden transition between normal and faulted behavior, figure 4.4 shows the result of gradually increasing D, similar to what would be the case for an actual degradation process.



(a) Two separate clusters

(b) Bimodal normal distribution

Figure 4.3: Left plot shows two distinct clusters: The major cluster represents errors associated to the estimation of the part of the vibration response which are caused by operating conditions. The smaller cluster to the right represent the vibration response being caused by degradation, which the model do not success to reproduce. This is confirmed by the rightmost part of the bimodal normal error distribution.

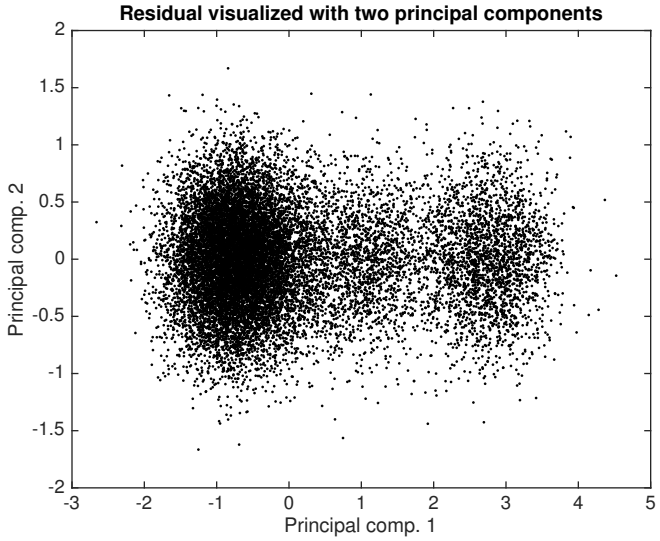
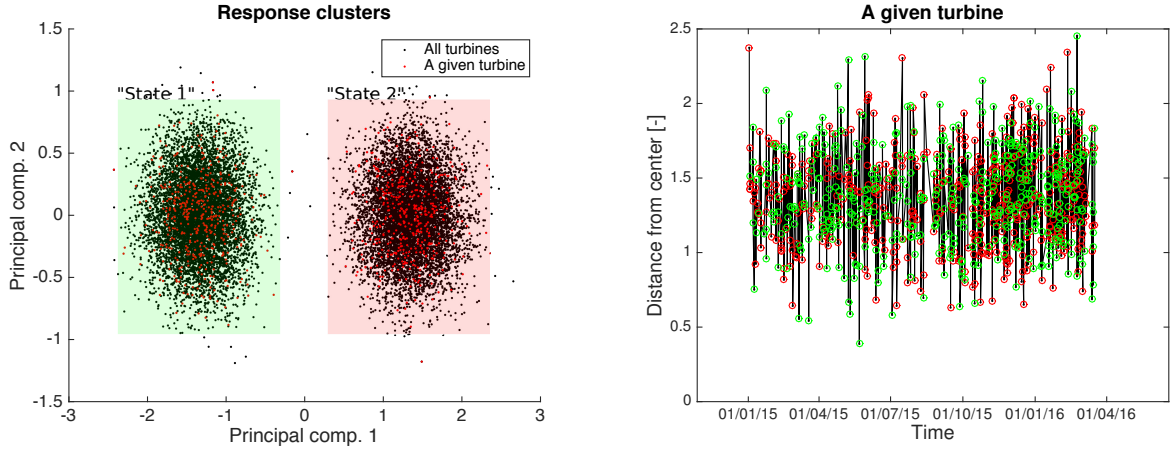


Figure 4.4: The figure shows two distinct clusters with intermediary points indicating a gradual transition between clusters.

To determine whether the clusters really represent degradation, i.e., that the minor cluster indeed represent a fault, a strong indicator would be how the transitions between clusters relates to time. In the case of rapid transitions where presence in the different clusters change during small time increments (days), degradation is not likely. To obtain this information, the distance from the major cluster representing normal behavior is monitored as a function of time. This is equivalent to monitoring the development of vibration amplitude as a function of time and serves therefor as a degradation monitoring technique. Figure 4.5 exemplifies the case of rapid transition between states. These types of time-invariant and rapid transitions between clusters would typically be a result of measurement noise or otherwise inconclusive phenomena. The plot in figure 4.5 was produced by assigning contribution from degradation to half of the observations in a randomized manner.

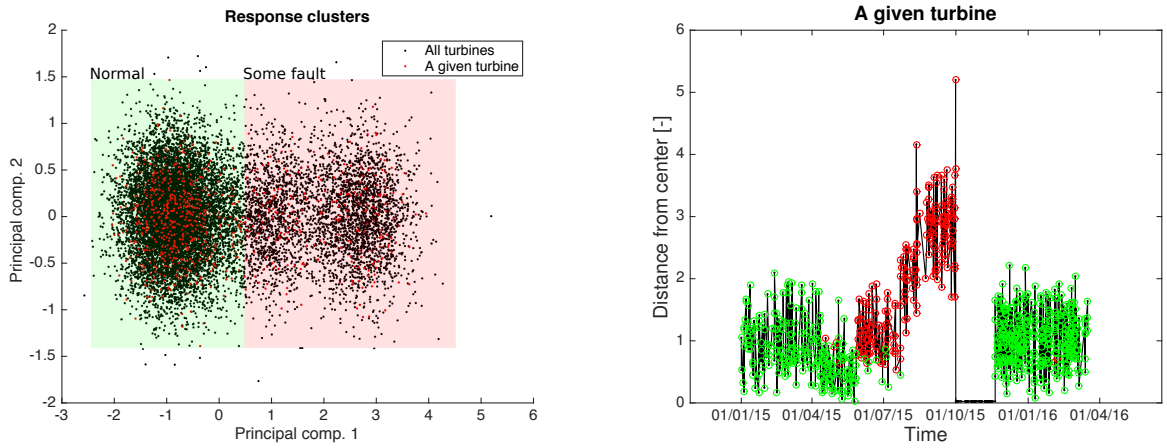


(a) Rapid transitions between clusters

(b) Shows no meaningful time-dependency

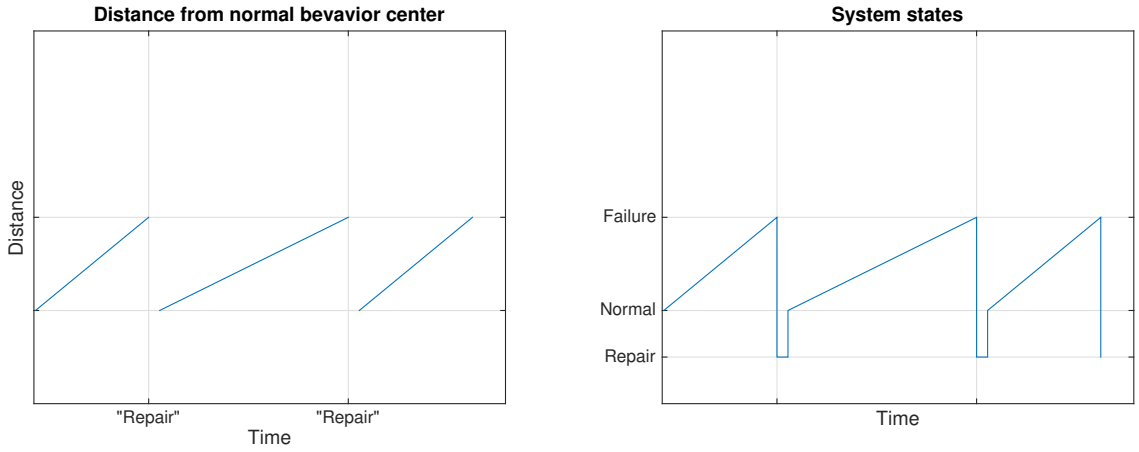
Figure 4.5: Figure illustrates the case where two distinct clusters (states) are present, but without an associated meaningful interpretation. The red and green markers are assigned to observations that are members of the left and right cluster respectively. The right plot shows rapid transitions between states.

The next scenario shows what happens when gradually increasing D as a function of time. The time-dependent increase in distance from the center of the major cluster representing normal behavior is now evident. A plausible interpretation from figure 4.6 would be an incipient fault originating in June 2015, before ultimately ending with a failure in October the same year. After two months downtime, the turbine is put back to normal operation. This if furthermore emphasized in figure 4.7 which relates the distance characteristic to possible equipment states.



(a) Gradually increasing vibr. amplitude **(b)** Time-dependent increase in distance

Figure 4.6: a) shows clusters being a result of gradually increasing D. b) shows that the scatter plot are representing fault evolution by relating the transition to time. The green and red makers are assigned according to the assumed value of principal comp. 1 with thresholds of < 0.5 and > 0.5 respectively.



(a) Characteristic behavior if degradation **(b)** Possible interpretations by states

Figure 4.7: a) shows a time-variant development of vibration amplitude which would be indicative of degradation. b) shows possible system states corresponding to a).

Finding patterns similar to what is shown in figure 4.7 can be valuable on several levels if auxiliary data such as fault logs were available to confirm the system states. Firstly one can support the rule or criteria for when to intervene with maintenance actions. For this to be efficient, information that relates which faults are associated with the observed clusters would be crucial. That way specific maintenance tasks could be scheduled. If furthermore a statistically sufficient number fault to failure progression timelines, i.e., the time period between "normal"

to "failure", were available, estimates of the remaining useful life could be provided.

The steps showed in the above use-case employs the very same principles as will be used to the real vibration data. Before we get there it is provided a formal introduction to the enabling techniques/tools enabling the approach. Associated MATLAB code for the use-case is found under `UseCase.m` in appendix B.2.1.

4.2 Selecting Techniques/Tools

The process of selecting tools can be described as 1) identifying what properties needs to be realized, and 2) to identify appropriate tools for their realization. Firstly, some way of validating the methodology assumptions needs to be in place. To do this, identifying patterns in the data could provide useful interpretations, making the need for visualization apparent. Secondly an appropriate model for vibration modeling needs to be chosen. To determine what model is appropriate, some way of measuring the model performances will be required. Ultimately this boils down selecting means to enable *visualization, vibration modeling* and a technique for *performance evaluation*.

4.2.1 Visualization

The frequency spectrum of the vibration data range from 0 to 8138 Hz and the resolution is 20,34. This makes a total of 401 frequency increment, or a *dimensionality* of 401 since each frequency increment is treated a random variable. To visualize this information, the need for dimensionality reduction becomes apparent. In the era of big data there exists a large multitude of techniques for reducing dimensionality, each with their strengths and weaknesses. As dimensionality reduction in many cases are used to reduce the computational burden by running algorithms with reduced data-sets, aspects concerning algorithm efficiency and accuracy are largely taken into account when evaluating the performance of the technique (Silipo, 2015). For this undertaking, the purpose is merely to visualize, hence making the choice more straight forward.

As such, the *principal component analysis* (PCA) is a well proven technique for dimensionality reduction. By choosing the number of principal components (dimensions) to include, the

user is able to manage how much of the data-set variance to include. PCA is furthermore used by Pozo and Vidal (2016) who successfully execute fault detection for wind turbines by comparing baseline PCA patterns representing healthy behavior to an inflow of current wind turbines with unknown status. This suggests that PCA indeed could be a viable way to visualize and explore the data and is hence chosen.

4.2.2 Generalization

Since the trustworthiness of the analysis relies heavily on choosing a model that makes a fairly correct mapping from input to output, a *set* of models rather than a single model is chosen. This is considered wise particularly due to the fact that the relationship between input (operating conditions) and output (vibration amplitude) in terms of linearity is yet unknown. Following this reasoning, three simple polynomial models of 1^{st}, 2^{nd} and 3^{rd} will be developed to check whether the performance is enhanced when the linearity increase. If so is the case, an appropriate model for mapping non-linear relations must be chosen.

When browsing the literature on PHM for wind turbines and in general, *two* methods are frequently reoccurring for mapping non-linear relations. That is, artificial neural networks (ANN) and support vector machine (SVM). ANNs has been extensively employed in numerous fields of science and technology, including prognostics and health management during the three last decades (Ata, 2015). SVMs, with a somewhat later entry into the PHM community, started to gain popularity in 1999 (Laouti et al., 2011). Both methods are used to understand complex non-linear relations based upon training with large amounts of data.

Ata (2015), who do a review on the use of artificial neural networks application for wind energy systems, groups the application of ANNs in three major categories: forecasting and prediction, prediction and control, identification and evaluation. Hush et al. (1997) who study the application of neural nets with special emphasis on vibration data for fault detection of faulty bearings conclude that the approach which presents the best probability of success is *trending*. Trending involves training the neural net on a well-behaving system. Faults are next detected by identifying deviations from the normal behavior benchmark.

Trending is furthermore used by Kusiak and Verma (2012) who analyze bearing faults in wind turbines from SCADA-data from 24 1.5 MW turbines collected over a period of four months by

deploying neural network algorithms to predict the normal behavior.

Yang et al. (2008) who presents a three layer neural network approach to diagnose the actual fault status of the gearbox, indicate in the their results that due to " excellent abilities of parallel distributed processing, self-study, self-adaptation, selforganization, associative memory and its highly non-linear pattern recognition it is an efficient and feasible tool to solve complicated state identification problems in the gearbox fault diagnosis".

In his doctoral thesis, Verma (2012) undertakes a data-mining approach utilizing SCADA-data and fault logs to identify critical turbine faults as well as predicting their occurrence. The work done is extensive and encompass identification of critical status pattern of wind turbines; prediction models for particularly fault prone components; anomaly detection based approach to analyze bearing overtemperature events; vibration analysis of the wind turbine gearbox as well as overall wind farm monitoring. For the segment concerning vibration analysis of the gearbox, ANN is chosen due to its well-known ability to approximate the non-linear relationship between input and output.

Laouti et al. (2011) are able to successfully detect and isolate faults within sensors and the generator by employing a SVM model for pattern recognition. Again, the underlying approach relies on a normal behavior benchmark which is compared to "real measurements" provided by a model created by Odgaard et al. (2013) that represents a three-bladed pitch-controlled variable-speed wind turbine with a nominal power of 4.8 MW.

Samanta (2004) who use vibration data from an experimental setup with normal and defective gears do fault detection by using both ANNS and SVMs. Both of the classifiers, which had their feature selected aided by genetic algorithms (GA), yielded comparable performance nearly 100 % even with different load conditions and sampling rates.

Widodo and Yang (2007) who presents a review about the use of SVM for fault detection claims that SVMs gives better generalization abilities than for instance ANNs for situations where the number of samples is scarce. Furthermore, they argue that through minimizing the upper bound generalization error (structural risk minimization) rather than minimizing the error on the training data set (empirical risk minimization), SVMs stand above its competitors. This aspects is mainly what differentiates ANNs and SVMs from each other. While the ANN is susceptible to identify a local optimum, the SVM guarantee the local and global optimal solution

are exactly the same (Vapnik, 1999)

As far as the literature discusses what is the better choice among ANN and SVM, the debate is related to the specific analysis at hand. This make sense because each of them can be trained and configured with a large variety of algorithms and architectures which makes them hard to compare in the general case. To really identify which method is the better choice, a basis for comparison should be formed by employing each of the methods to solve the same specific problem. For this project it has not been prioritized to employ both techniques. But since there is no definite rationale to choose the one over the other, the ANN is chosen due to being a well-proven technique for mapping non-linear behavior. Moreover, as NTNU students have access to MATLAB, is it convenient that ANNs are supported by the Netural Network Toolbox ™ which is part of the MATLAB functionality.

4.2.3 Performance evaluation

To select the best candidate from the set of models, we need some way to evaluate the mutual model performances. A well-proven technique as such is k-fold cross-validation Walpole (2012). Typical applications are Leek (2013): i) picking which variables to include in an analysis; ii) picking the type of predictive function or iii) picking parameters for the predictive function. In this case there is a match with ii) as the objective is to choose the better performing model. For a regression problem like this, it is agreed upon by multiple sources (e.g, see Arlot and Celisse, 2010; MATLAB, 2015) that a standard cross-validation technique like k-fold cross validation would provide a robust estimate for the model performance. k-fold cross validation is hence chosen.

4.3 Formal Description of Techniques/Tools

4.3.1 Principal component analysis

Aldrich and Auret (2013) express the aim of the principal component analysis (PCA) as "...to define a set of principle components consisting of linear combinations of the original measurement variables such that the first principal component accounts for the most variance in the data set, the second principal component for most of the remaining variance, etc." More

formally, when $k < n$, the PCA algorithm conducts a mapping from an n-dimensional feature space to a k-dimensional feature space ($\mathbf{x} \in \Re^n \rightarrow \mathbf{z} \in \Re^k$). Ng (2016) explains the involved steps as follows:

1. Normalize the data: For the data set $X = \{x^1, x^2, ..., x^m\}$ of m columns-wise variables, calculate the population mean (μ) and standard deviation (σ) and replace x_j^i with $\frac{x_j^i - \mu}{\sigma}$.

2. Compute the co-variance matrix, Σ

$$\Sigma = \sum_{i=1}^{n} (x^{(i)})(x^{(i)})^T, \tag{4.6}$$

3. Compute the eigenvectors matrix of Σ.

4. The matrix made up by the k first eigenvectors of Σ, denoted U_{reduced}, is next used to obtain the k-dimensional feature space $\mathbf{z}$ by the following operation: $\mathbf{z} = {C_{\text{reduced}}}^T \cdot X$.

The result from the transformation is shown in figure 4.8. The first principal component has the largest possible variance (middle) whereas the succeeding component (left) has the highest possible variance under the constraint that it is orthogonal to the preceding component.

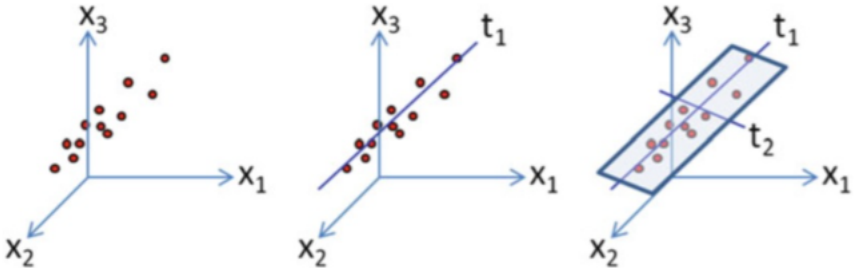


Figure 4.8: Left: scatterplot; middle: first principle component (t_1); right: first and second principle components (t_1 and t_2). Figure from Aldrich and Auret (2013).

4.3.2 Polynomial models

Equation 4.7-4.9 represents the single variable-case for the 1^{st}, 2^{nd} and 3^{rd} degree polynomials respectively

$$\hat{y} = a_1 \cdot x_1 + a_0 \tag{4.7}$$

$$\hat{y} = a_2 \cdot {x_2}^2 + a_1 \cdot x_1 + a_0 \tag{4.8}$$

$$\hat{y} = a_3 \cdot {x_3}^3 + a_2 \cdot {x_2}^2 + a_1 \cdot x_1 + a_0 \tag{4.9}$$

where $\hat{y}$ is the estimate of y. The model error is represented by ε making $y = \hat{y} + \varepsilon$. Furthermore, a_0 is a constant and a_1, .. , a_3 are coefficients determined by ordinary least square criterion whose optimum is achieved when the sum, S, of the squared residuals, $\varepsilon_i = (\hat{y} - y)^2$ is a minimum:

$$\min S = \sum_{i=1}^{n} \varepsilon_i$$

As for the current analysis there is 401 frequency increments, each of which are having their associated vibration amplitude estimated. This is taken into account by the generalized formulation in equation 4.10 allowing for m values for each of the n variables.

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} 1 & x_{1,1} & x_{1,2}^2 & \cdots & x_{1,n}^n \\ 1 & x_{2,1} & x_{2,2}^2 & \cdots & x_{2,n}^n \\ 1 & x_{3,1} & x_{3,2}^2 & \cdots & x_{3,n}^n \\ \vdots & \vdots & \ddots & \vdots & \\ 1 & x_{m,1} & x_{m,2}^2 & \cdots & x_{m,n}^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \vdots \\ \varepsilon_m \end{bmatrix} \tag{4.10}$$

4.3.3 Artificial neural networks

The inventor of one of the first neurocomputers, Dr. Robert Hecht-Nielsen defines an artificial neural network as " ... a computing system made up of a number of simple, highly interconnected processing elements, which process information by their dynamic state response to external inputs." This computing system is inspired by studies of the human's central ner-

vous system, where the interaction between nodes are to mimic the brains neurons and their synaptic connections. As opposed to the polynomial fitting, the mathematics involved with an artificial neural network is less trivial. The underlying principle are however the exact same as for prediction problem, namely to substantiate for the best generalization by minimizing the generalization error, ϵ.

Among the many types of ANNs, the multilayer perceptron (MLP) neural networks has gained the most popularity in engineering applications (Yang et al., 2008; Ata, 2015). This network type is also used in this work. MLPs belongs to the class of *feed forward neural networks*, meaning that multiple layers of nodes are configured in a directed graph. A typical neural network architecture for such a network is shown below. n, q and m denote the number of neurons in the input, - hidden - and output-layer respectively.

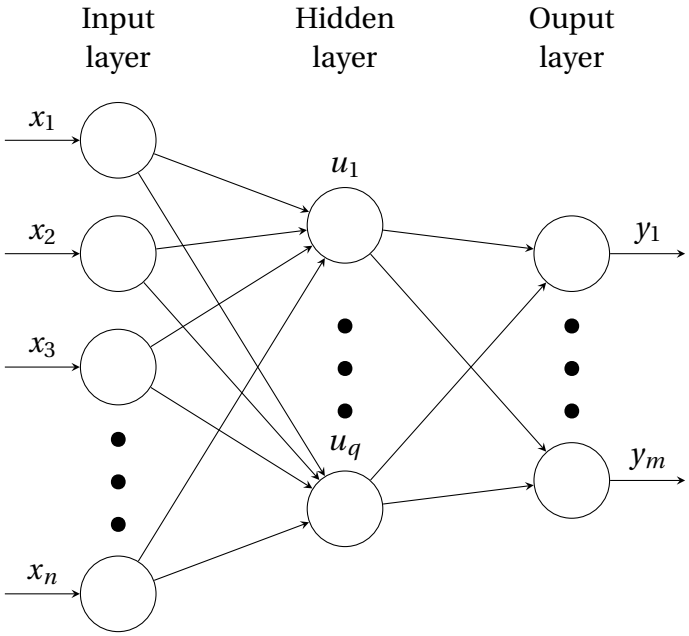


Figure 4.9: Artificial neural network structure.

The input layer is where the input goes, and is hence uniquely defined once the shape of the data-set is known. This is also the case for the output layer, which for the current analysis is *one*. This is because 401 networks will be created - one for each frequency increment.

What determines the estimated output is the *weights* attributed from each of the inputs. These are calculated within the hidden layer, where the actual processing take place. Hence,

the key idea in training a neural network is to determine the *connections weights* that minimize the error function between the network output and the corresponding target values from the training set. This is most commonly done through *back-propagation*, which is further described in the below work-flow:

Step 1: Each input node sends out their values towards the first echelon of neurons in the hidden layer. For the first iteration the connection weights are random, as learning has not yet occurred (Suarez, 2009).

Step 2: To determine the connection weights to be carried over from each of the input nodes, each hidden layer neuron is equipped with a transfer function - often the Sigmoid function: $\phi(v) = \frac{1}{(1+e^{-v})}$ (Hastie et al., 2009). As a consequence, the input stimuli might not be accepted at all. Usually though, the transfer function will return a value that is a combination of the current node's value and the trigger value(s) (Suarez, 2009).

Re-entering the loop/Validation stop: From the get-go, connection weights are random. For each time output that is calculated, the weights are re-calibrated by evaluating the model error. This process is known as back-propagation. How the back-propagation specifically is carried out would depend on which training algorithm is used but would in any case intend to minimize the network error, ε. Depending on the stop criteria, the process is either repeated or stopped. Typically training continues until validation error fails to decrease for a number of iterations (MathWorks, 2016).

More formally, and based upon Samanta (2004), the neural network operating principle can be formalized in mathematical terms. The input vector, $\mathbf{x} = (x1\ x2, ..., x_n)^T$ is transformed to an intermediate vector of hidden variables $\mathbf{u}$ through the activation function ϕ_1. The output from the jth node in the hidden layer, u_j, is calculated according to:

$$u_j = \phi_1 \left(\sum_{i=1}^{n} {w^1}_{i,j} + {b^1}_j \right) \tag{4.11}$$

where ${w^1}_{i,j}$ and ${b^1}_j$ denote the weight of the connection weights between the jth node in the hidden layer and the ith input node. A superscript of 1 represent the first connection (between the input and the hidden layer, 2 would represent the next connection etc. The output

vector $\mathbf{y} = (y1, y2, ..., y_m)^T$ is obtained from the vector of intermediate variables $\mathbf{u}$ in an analogous manner using the activation function ϕ_2. The output of the kth neuron is then expressed as:

$$y_k = \phi_2 \left(\sum_{i=1}^{q} {w^2}_{l,k} + {b^2}_k \right) \tag{4.12}$$

The superscript of 2 is now representing the second connection between neurons in the hidden - and output-layer. Together with the Sigmoid function, the hyperbolic tangent and the peace-wise linear functions given by equation 4.13-4.14 are well-proven candidates for calculating the connections weights. Interested readers are referred to e.g. Kriesel (2005); Du and Swamy (2014) for more on the topic.

$$\phi(v) = \frac{1 - e^{-2v}}{1 + e^{-2v}} = \frac{2}{1 + e^{-2v}} - 1 \tag{4.13}$$

$$\phi(v) = v \tag{4.14}$$

To optimize the network topology, i.e., the number of neurons in the hidden layer, the arising trade-off between training - and validation error must be considered. As indicated in figure 4.10, the training error becomes smaller and smaller as more neurons are added. The validation error does on the other hand experience a turning point - from which increasing validation errors results. This is a consequence of the potential over fitting which has occurred. Thus, ensuring an optimized network topology typically relies on iterations revealing the point corresponding to optimal performance Lefebvre (2016). This is indicated with a cross in figure 4.10.

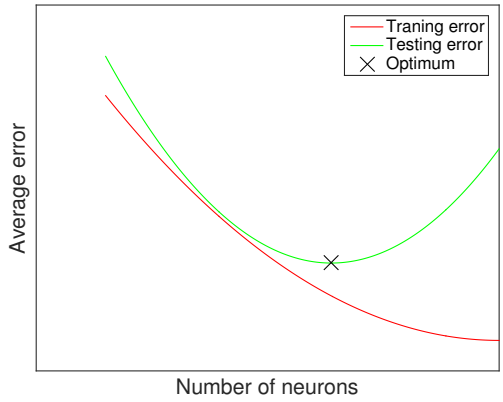


Figure 4.10: Tradeoff between training error and validation error. Optimum indicated by cross.

There are made efforts to overcome the tedious task of optimizing through trial and error. Liu et al. (2013) is able to quickly identify the optimum network parameters by using genetic algorithms (GA) and particle swarm optimization (PSO). The same source also finds that the following equation tend to define the range within which the optimum number of neurons in the hidden layer is present:

$$\text{No. of hidden neurons} = \sqrt{m+n} + a \quad a \in [1,10] \tag{4.15}$$

where m and n is the number of input and output neurons respectively a is an adapting variable.

The feed-forward MLP network used in this work have a three-layer architecture similar to the one shown in figure 4.9. It was created, trained and implemented using the Neural Network Toolbox™ which is a Matlab application. Training was enabled with the Lavenberg-Marquardt algorithm which iteratively performs back-propagation to minimize the mean squared error of the discrepancy between network output and target values (MathWorks, 2016).

4.3.4 *k*-fold cross-validation

k-fold cross-validation partitions the training data into k equally sized subsets where a single subset is retained as the test set and the remaining nine subsets are used as training data. Figure 4.11 illustrates the partitioning for the case where k is 10.

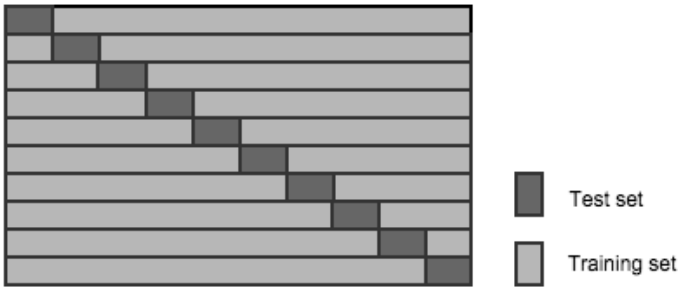


Figure 4.11: Training data partitioned into 10 equally sized subsets, allowing for 10 different cross-validations (folds) so that each of ten subsamples are used *once* as validation data. Figure inspired by Pereira et al. (2009).

For a k of 10 the cross-validation process is repeated 10 times so that each of the ten subsamples are used exactly once as validation data. These repetitions are known as *folds*. For each fold the following steps take place Leek (2013): 1) build a estimating function based on the training-data; 2) evaluate on the test set; and 3) repeat, or if at last fold: average the estimated errors. Among several options for calculating the error, mean squared error (MSE) is chosen as it incorporates both bias and variance. MSE is calculated according to equation 4.16.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2 \tag{4.16}$$

where n denotes the number of predictions .$\hat{Y}$ and Y is the predicted and true values respectively.

4.4 Constructing a Condition Monitor

The methodology can be divided into the following steps: 1) Filtering and pre-processing; 2) generalization and model evaluation development of model(s) and 3) residual analysis. The contents and associated output from each of the step is shown in figure 4.12. Relevant MATLAB-code from this chapter are given by `Filtering.m` and `MainAnalysis.m` found in appendix B.2.2 and B.2.3 respectively.

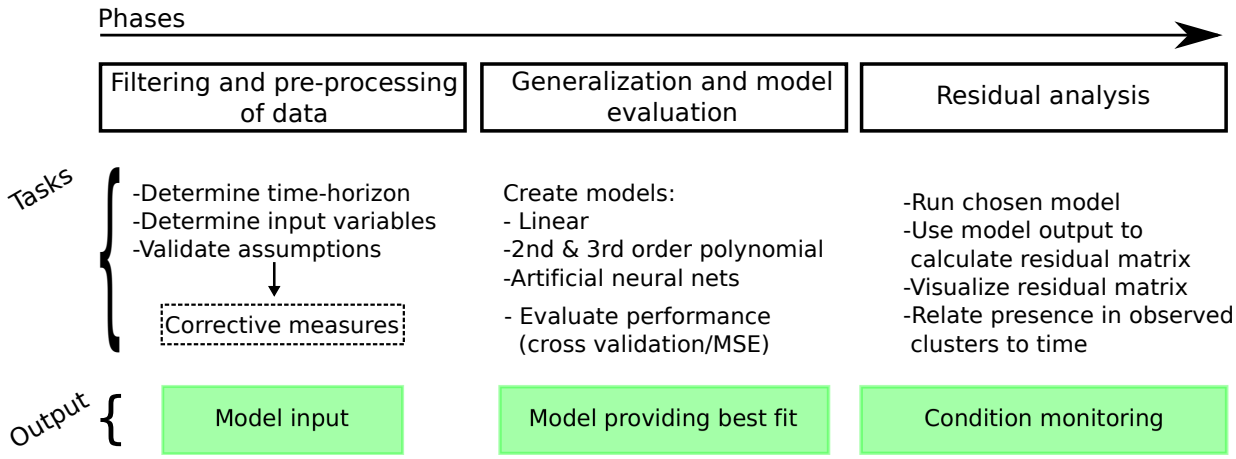


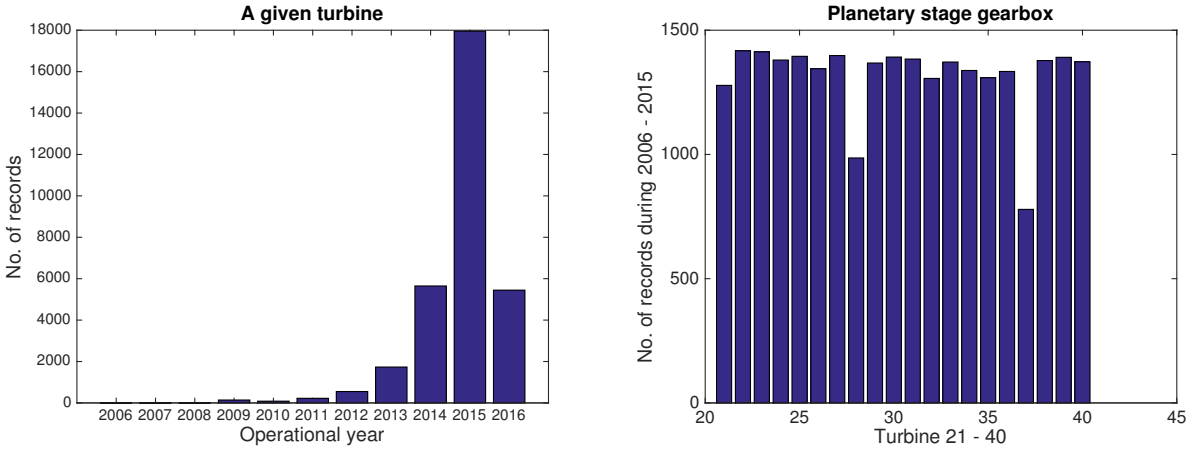
Figure 4.12: The three steps of the analysis associated with tasks and output.

4.4.1 Filtering and pre-processing of data

This phase involves three work packages: The delimitation of an appropriate time-horizon, assumption validation and lastly the identification of appropriate model-input.

Defining a time-horizon

Recalling from section 3.2, most records are made in recent years. The annual number of records for the planetary stage of the gearbox is shown in the left plot in figure 4.13. As of 2016, the most recent record is from 16^{th} of March, meaning that almost the first quarter of 2016 is covered. Maintaining this amount of data-capture throughout the entire year of 2016 would make the frequency of data-capture approximately four times as extensive as in the year of 2014. However, when comparing 2016 to the year of 2015, the capture-rate seem to be somewhat in the same range. Although no guarantee is granted, similar capture-rates could be indicative on similar data-capture criteria, which would be a prerequisite for obtaining a consistent data-base.



(a) Annual No. of records

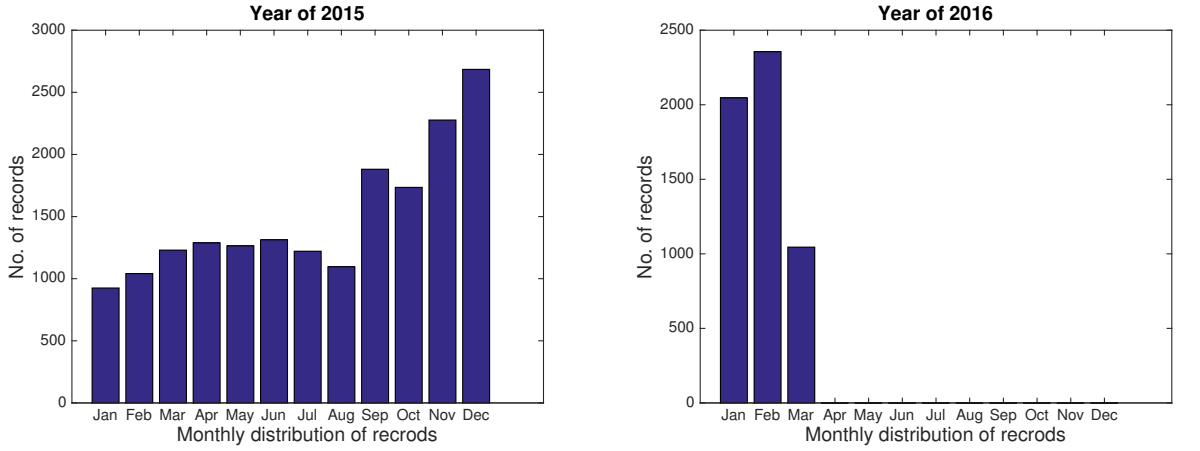
(b) Record distribution by turbine (T21-T40)

Figure 4.13: Left plot shows that more records stems from recent years. Right plot shows that records are quite evenly distributed among the turbines. Exceptions from this is observed for turbine 21, 28 and 37.

Figure 4.13 b) is made to reveal whether records are distributed evenly among the different turbines. Although turbine 21, 28 and 37 are slightly underrepresented, an even distribution seem to be the case otherwise. This is convenient, as large variations could be indicative of different data-pull settings across the turbines, making it problematic to evaluate the entire wind farm at once.

To challenge the assumption on similar data-capture regimes further, the monthly distribution of records for the year of 2015 and 2016 are shown in figure 4.14. The figure shows a significant increase in data-capture in the late part of the year as compared to the preceding months. This might be due to the fact that weather is harsher during the winter, which in turn would call for a more rapid data-capture as the load makes the turbines more susceptible to degradation. The same relative pattern is found for the year of 2014, although the absolute number of records is lesser.

When following the rationale outlined above, namely that similar data-pull rates could indicate similar data-pull criteria, the year of 2014 is excluded from the further analysis. Thus, records stemming from the time period January 2015 - 16^{th} of March 2016 will be considered for the further analysis.



(a) Records per month in 2015

(b) Records per month during Jan-March 21^{th} of 2016

Figure 4.14: The figure shows that more records are made during the winter as opposed to the remaining part of the year.

Validating assumptions through visualization

An initial overview of the vibration-data is sought trough visualization. If for instance the data-points within the same power bin fall close together, it would be consistent with assumption I stating that most of the data is normal. Furthermore, distinct clusters being formed as a function of rated power would suggest that the operating conditions indeed plays a role in determining the vibration response.

First off, the vibration response yielded from all turbines are plotted together for each of the power bins. Using two principal components, figure 4.15 reveals that similar patterns can be observed for all bins except the lowest power-bin ranging from 0 - 920 kW. The deviating pattern observed in this bin may be explained party from the wide range of power output contained in it. But perhaps most importantly the fact that this bin contains zero output turbines makes it susceptible to showing different behavior. To avoid having the data-base polluted from the diversity from the lowest bin it is discarded from the data-set.

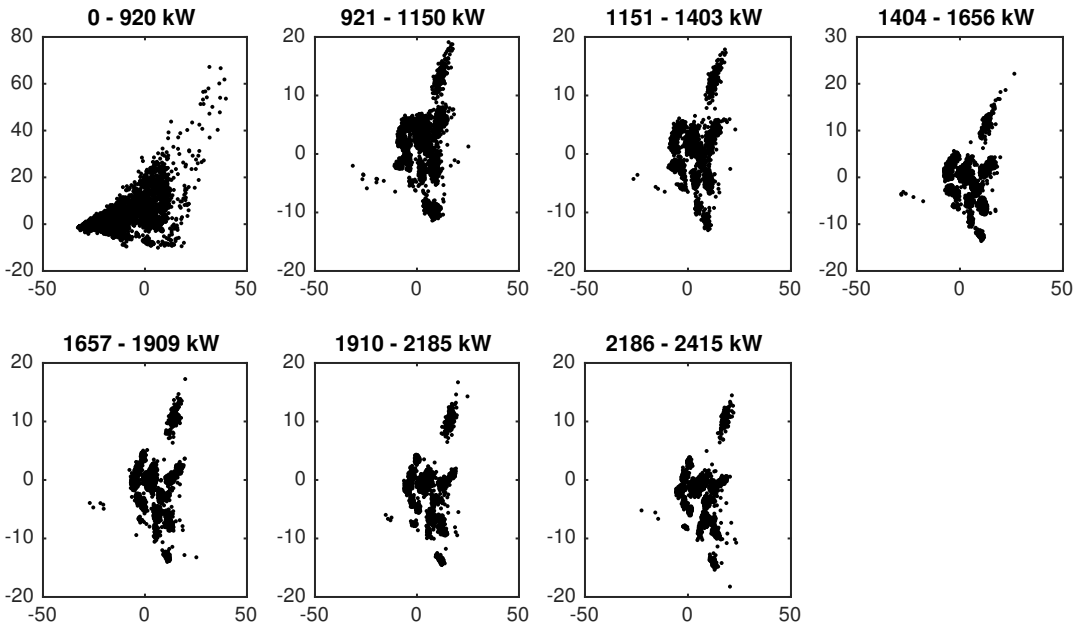


Figure 4.15: Vibration response per power bin when all turbines are considered together. Arguably due to step-effect from wide range of power outputs including the zero-output case, the 0-920kW bin shows a deviating pattern.

At first glimpse, the clusters were believed to correspond to different equipment status, but as shown in figure 4.16, each turbine give rise to their specific cluster. Only the vibration response from turbine 21-24 is plotted in the figure, but a confirmation of the finding was made by doing similar plots for the entire set of turbines. Thus the second hypothesis assuming that all turbines can be considered alike is compromised. To handle this rather unexpected finding, two possible ways to continue the analysis has been identified. Either the turbines are considered separately, each with a dedicated condition monitor; alternatively all turbines are considered together, but after some normalization procedure correcting for their relative offsets. The latter option is chosen as it simply would be more convenient to have *one* condition monitoring regime applicable to all turbines.

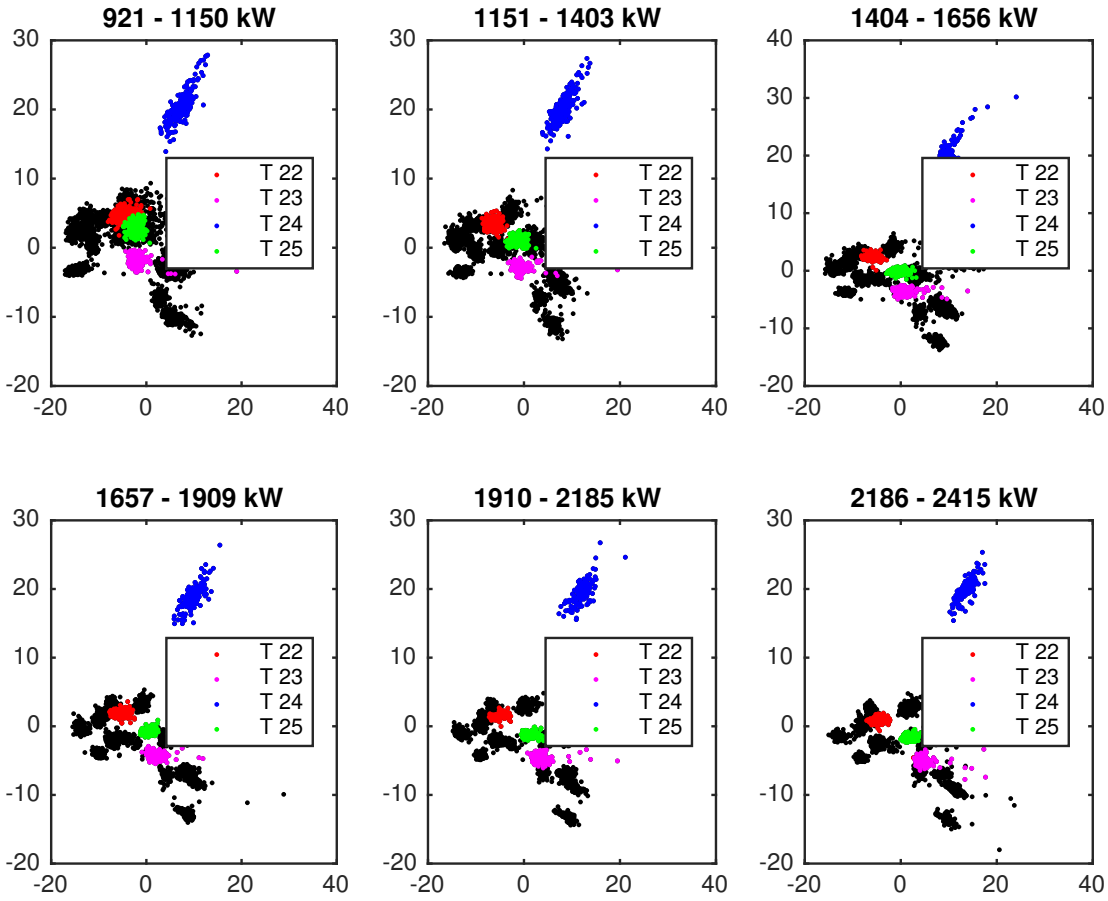
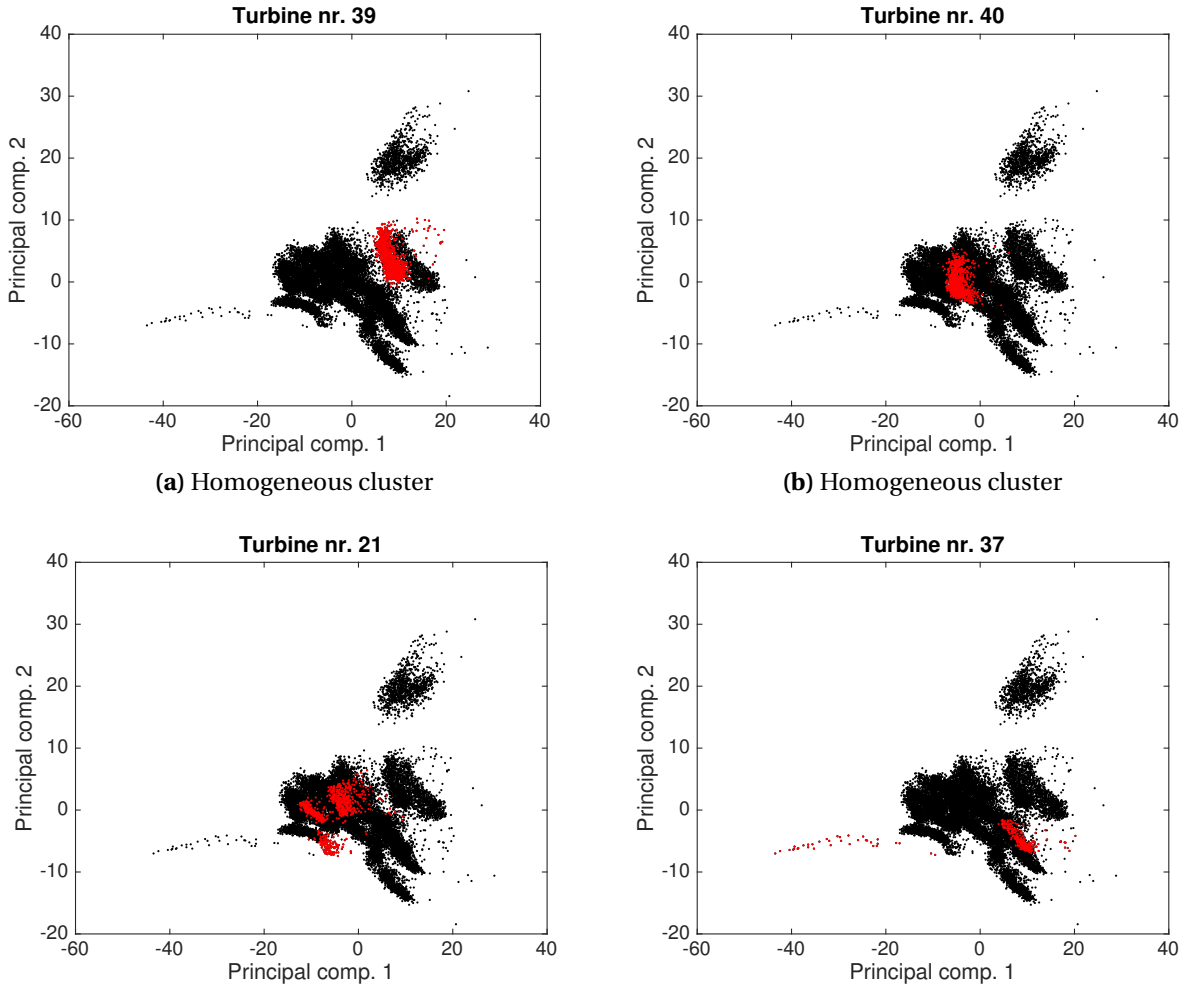


Figure 4.16: Response for selected turbine show turbine-specific behavior.

The data-set contains records for turbine 21- turbine 40, i.e. 20 turbines. For the continued analysis, turbine 21 and 37 are discarded from the data-base due to deviating behavior - again, to avoid pollution from deviating behavior. Their response is shown in figure 4.17 together with turbine 39 and 40 who represent the response characteristic formed by the remaining turbines.



(a) Homogeneous cluster

(b) Homogeneous cluster

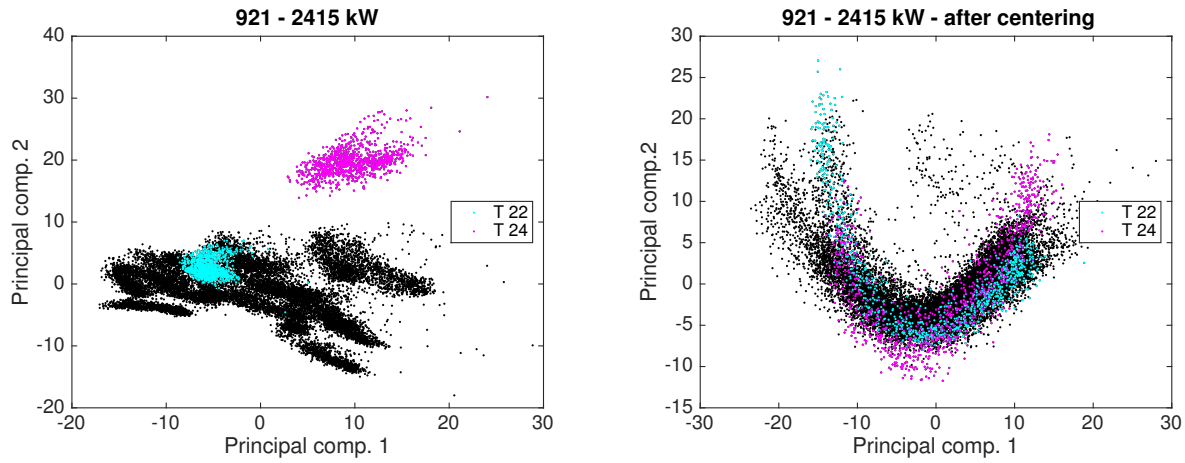
(c) Multiple clusters

(d) Strange behavior

Figure 4.17: Most turbines form response-clusters that are relatively homogeneous, such as a) and b). c) and d) deviates considerably and is therefore discarded from data-set.

Correcting for offset between the different turbines

The offsets are corrected for using a simple bias correction technique where each turbine response are centered according to their median value. Being less sensitive to outliers, the median is chosen over the mean which could be an alternative solution. After removing turbine 21 and 37, both polluting the data-set with uncharacteristic behavior, figure 4.18 shows the remaining data-set in a pre/post correction manner.



(a) No correction. **(b)** After Correction

Figure 4.18: Left plot shows clusters before the data set is adjusted for turbine-specific behavior forming clusters. Right plot depicts the resulting responses after median-centering the turbine responses.

After making all the turbines have the same median value, the turbine specific behavior is significantly less distinguishable, although still possible to recognize. Figure 4.18 shows turbines 22 and 24 which is fairly representative for the offsets that still is present between the turbines. The degree of distinctness between each turbine is however considered small enough to proceed the analysis assuming hypothesis II is still valid: All turbines can be considered equal. What remains to be investigated is how the operating conditions impact the vibration response. To do this, the vibration response as function of the power bin is plotted. The resulting plots in figure 4.19 shows that distinct clusters arise for each power-bin. Although this was expected, it nevertheless is an important finding because the modeling taking place later on relies heavily on the fact that rated power is impacting the vibration response.

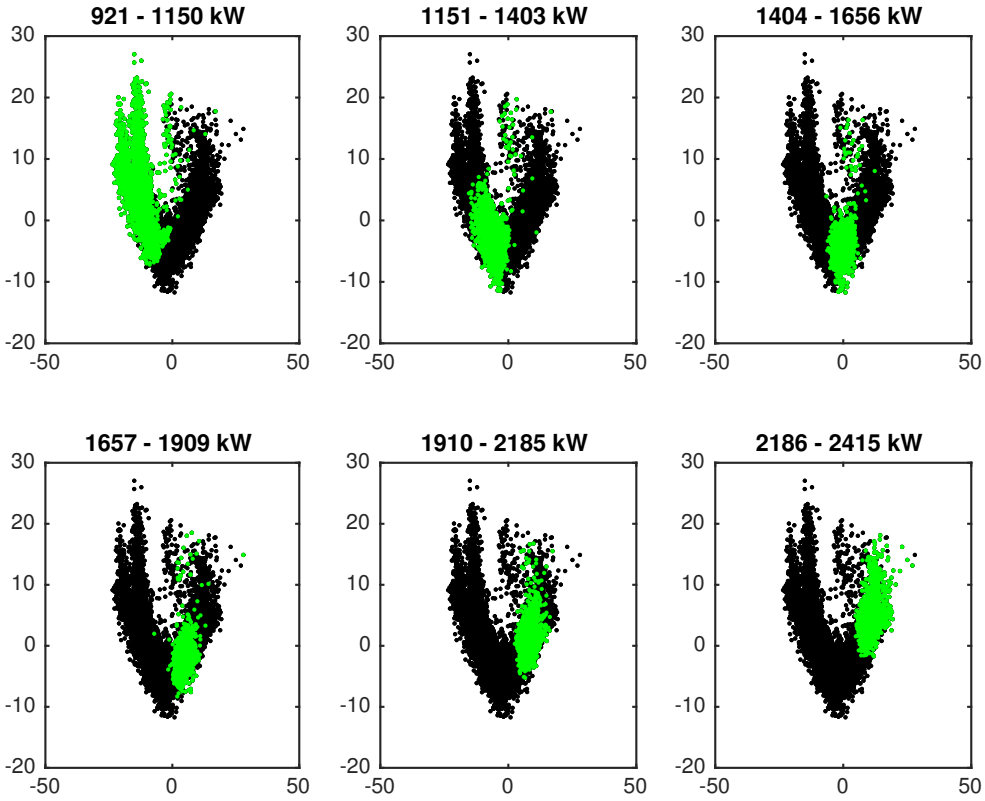


Figure 4.19: Turbine responses per power-bin forms distinct clusters for different rated power confirming the correlation between power and vibration response.

Identifying appropriate co-variables for model training

The last step of the pre-processing phase involves identifying which operating conditions the models shall be trained with. Intuitively, and because the literature agree on it (e.g., see He et al., 2016; Antoniadou et al., 2015), the vibration response is partly driven by the load experienced by the turbine. This was furthermore confirmed in figure 4.19. Among the operating data being available in the data-set which is reintroduced in table 4.1, *active power, wind speed and yaw angle* are the ones found reasonable to impact the vibration response. Since it is shown that the rated power has a clear impact on the vibration response (see figure 4.19), the associated impacts from wind speed and yaw angle can easily be indicated by plotting their correlation to the average active power. The resulting plots are shown in figure 4.20.

Table 4.1: Reintroduced: Table showing what information is stored in a record and how it is structured.

Col. 1	Col. 2	Col. 3	Col. 4	Col. 5	Col. 6	Col. 7	Col. 8	Col. 9	Col. 10	Col. 11	Col. 12-412.
Turbine number	Rec. number	Power-bin (lower)	Power-bin (upper)	Power (avg.)	Yaw-angle	Rotor RPM	Wind speed	Gen. RPM	RPM	Date/ time	Feature measurements

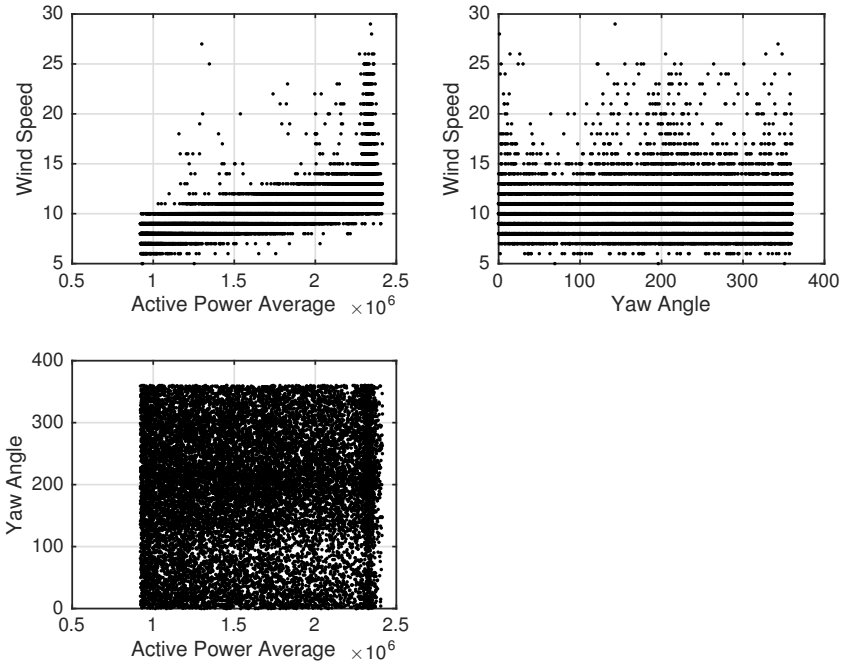


Figure 4.20: Correlation can be seen between wind speed and active power. This is not the case for the yaw angle.

When considering the relationship between wind speed and active power a quite conclusive correlation is observed. From a common sense point of view this could be expected. The yaw angle seems however not to be correlated to the wind speed, and consequently nor to the active power. Hence the co-variables *rated power* and *wind speed* will be used for model-training.

Summarizing the filtering and pre-processing phase

The intention of this phase was to pre-process and filter the data to obtain an appropriate data-set for model-training. In brief, the following important actions has been taken:

- Due to similar capture-rates, only records collected during January 2015 - 16^{th} March is considered.

- As the lowest power-bin showed strange behavior it was discarded from the data-set. Turbine 21 and 37 was discarded for deviating significant from the other turbine responses.
- Since each of the turbines are forming distinct clusters, the vibration response was median-centered to allow for *one* condition monitoring model applicable to all turbines.
- Appropriated input-variables for model training are identified to be active power and wind speed.

With respect to assumption I, most of the data seem to fall according to *one distinct cluster* which is consistent with most of the data being normal. Assumption II stating that all turbines can be considered alike is wrong. However, after the correction, the offsets are considered so small that this effect can be neglected. Regarding assumption III, a clear correlation is found between rated power and vibration response.

4.4.2 Model development and evaluation

Four models are developed, each with a given level of linearity:

i. A linear regression model, i.e., a polynomial of 1^{st} degree

ii. 2^{nd} and 3^{rd} degree polynomial models

iii. Artificial Neural Networks

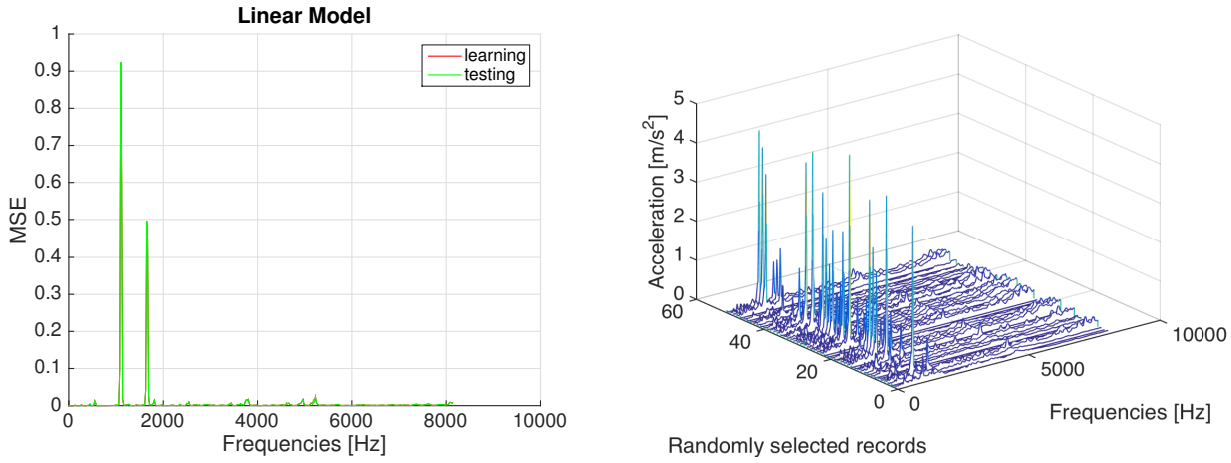
The input/output - configuration is the same for all models and will hence be described right away to cover the forthcoming model developments. After pre-processing a total of 16818 records form the basis for model training. With reference to the below matrices, $X_{i,j}$ represent the input data and $Y_{i,j}$ represent the targets. Within $X_{i,j}$, the j columns represents rated power and yaw angle, thus making n equal to 2. Each of the i rows represent records, making i go through $1, 2, ..., 16818$. With a frequency spectrum ranging from 0-8.138 kHz and a resolution, Δf, of 20,34 Hz, 401 variable results. Thus, the number of columns in the target matrix, k is equal to 401. All 401 variables are not estimated simultaneously, but one at a time using dedicated sub-models. This yields 401 sub-models, that together form the complete model covering the entire frequency spectrum.

$$X_{i,j} = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,n} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m,1} & x_{m,2} & \cdots & x_{m,n} \end{bmatrix}, Y_{i,j} = \begin{bmatrix} y_{1,1} & y_{1,2} & \cdots & y_{1,k} \\ y_{2,1} & y_{2,2} & \cdots & y_{2,k} \\ \vdots & \vdots & \ddots & \vdots \\ y_{m,1} & y_{m,2} & \cdots & y_{m,k} \end{bmatrix}$$

Furthermore, to evaluate model performances, all models are subjected to k-fold cross validation with k=10 and mean squared error (MSE) as the performance metric. When the goal is model selection it is often reported that the optimal k is between 5 and 10, because the statistical performance does not increase a lot for lager values of k, and averaging over less than 10 splits remains computationally feasible (Hastie et al., 2009, Section 7.10). The training/testing-ratio is set to 70 % / 30 % for all models.

Linear model performance

The overall MSE, that is, the average squared error across all frequencies for the linear model is 0.00855 for training and 0.00856 for testing. The closeness in values between training and testing becomes apparent in figure 4.21 a) showing the MSE for each of the 401 variables. The sudden jumps along the y-aksis are manifests of the fact that the vibrations tend to have a vibration peaks around this region. This is illustrated in 4.21 b). Missing the peak values then gives relatively large contribution to the overall model error.



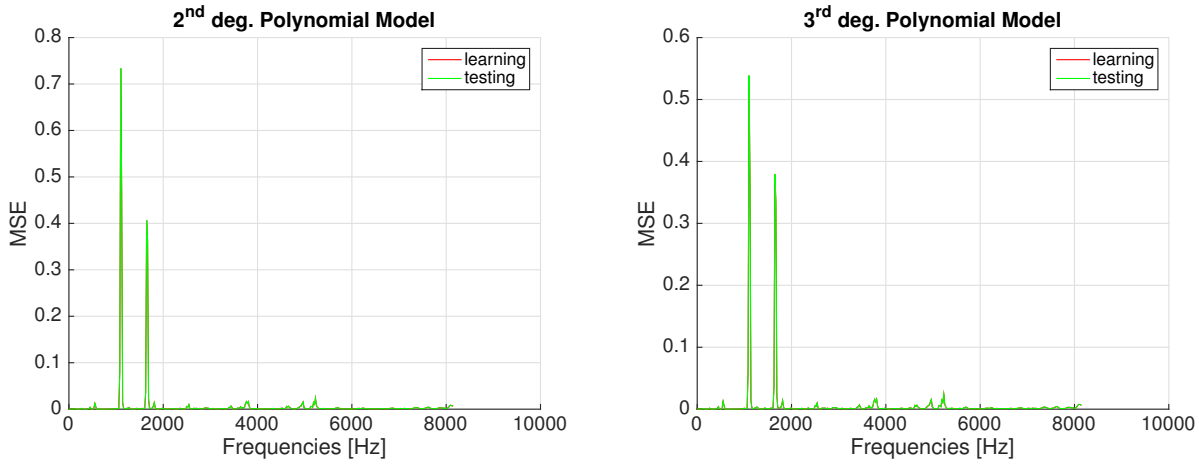
(a) MSE from linear model

(b) Frequency domain vibration response

Figure 4.21: Average MSE for training and testing for each frequency increment.

2^{nd} and 3^{rd} order polynomial model performances

The overall MSE for the 2^{nd} order polynomial was 0.00713 and 0.00714 for training and testing respectively. Associated average MSEs for the 3^{rd} polynomial, was 0.00617 and 0.00618. Again it can be observed clear jumps at the frequency around the peaks.



(a) MSE 2^{nd} order polynomial

(b) MSE 3^{rd} order polynomial

Figure 4.22: Average MSE for training and testing for each frequency increment.

As a curiosity, and to furthermore validate the correlation between input and targets, the 3^{rd} order polynomial model is run with each of the input variables separately. This was also done for the yaw angle and ultimately for a "random variable". For the random variable run, the input are randomized so they no longer match the targets. The two latter runs show that indeed the yaw angle is poorly correlated to the vibration response, as the resulting performance is no better than for the random variable.

Table 4.2: Considering each input individually to indicate degree of correlation.

	Model input			
MSE for training/testing:	Rated power	Wind speed	Yaw angle	Random Variable
Average MSE learning	0.00618	0.0102	0.0134	0.0135
Average MSE testing	0.00619	0.0103	0.0135	0.0135

Neural network performance

Recalling (section 4.3.3) that the network topology encompass the determination of number of neurons in the different layers. The input and output layer have their number of neurons uniquely defined by the data-set. That is, *two* neurons in the input-layer corresponding to the rated power and the yaw angle, and *one* neuron in the output layer corresponding to the variable being estimated. Analogously as for the previous models, the complete neural net model consists of 401 nets - one net per variable. Ideally, the number of neurons should be optimized for each of the submodels. Due to the considerable computational time required to train the nets, optimizing 401 networks has not been prioritized in this project. A compromise was however made by training all of the networks, i.e., the global model with two different hidden layer configurations. As a starting point, the relation of Liu et al. (2013) (No. of hidden neurons $= \sqrt{m+n}+a \quad a \in [1,10]$) was used to identify 10 hidden neurons as a reasonable starting point. The model was in turn trained with 20 hidden layer neurons. As the latter provided a slightly better fit, the 20 neuron network was chosen. The resulting network topology is shown below.

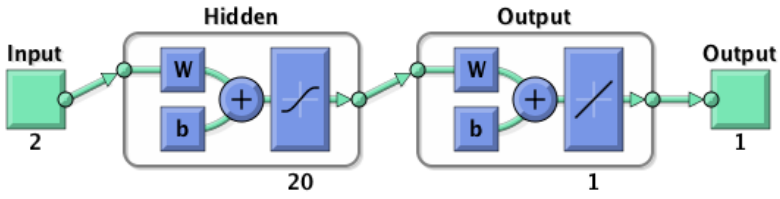


Figure 4.23: Overall architecture of the neural networks.

For the 10 neuron case, the overall MSE was 0.00562 for training and 0.00564 for testing. The 20 neuron case performed slightly better at generalization with a test-set error of 0.00557. The distribution of error across the frequency increments follows the same pattern as for the other models, but is less severe. Therefore the 20 neuron neural networks are chosen as the final model. Figure 4.25 shows the learning curve from one of the networks. For every iteration, the model error gets lower. Training stops when 6 subsequent iterations has failed to further decrease the validation error.

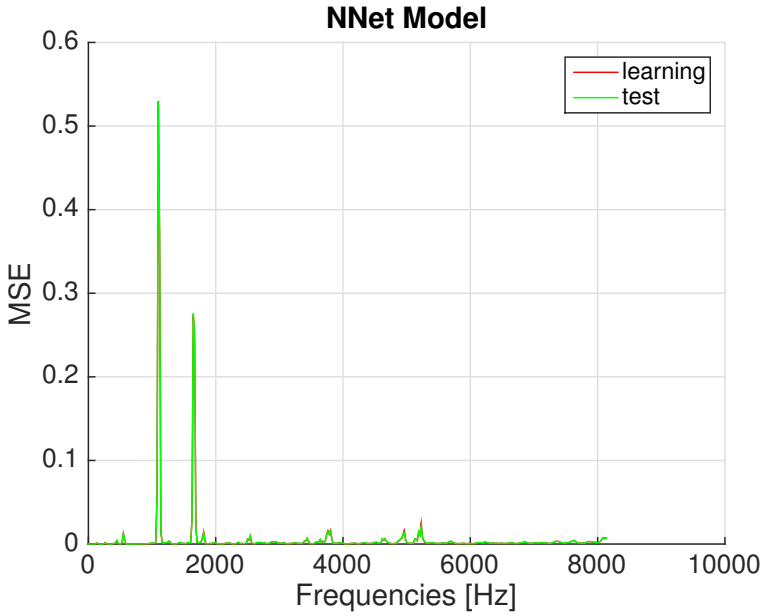


Figure 4.24: MSE for each frequency increment for 20 neuron hidden layer case.

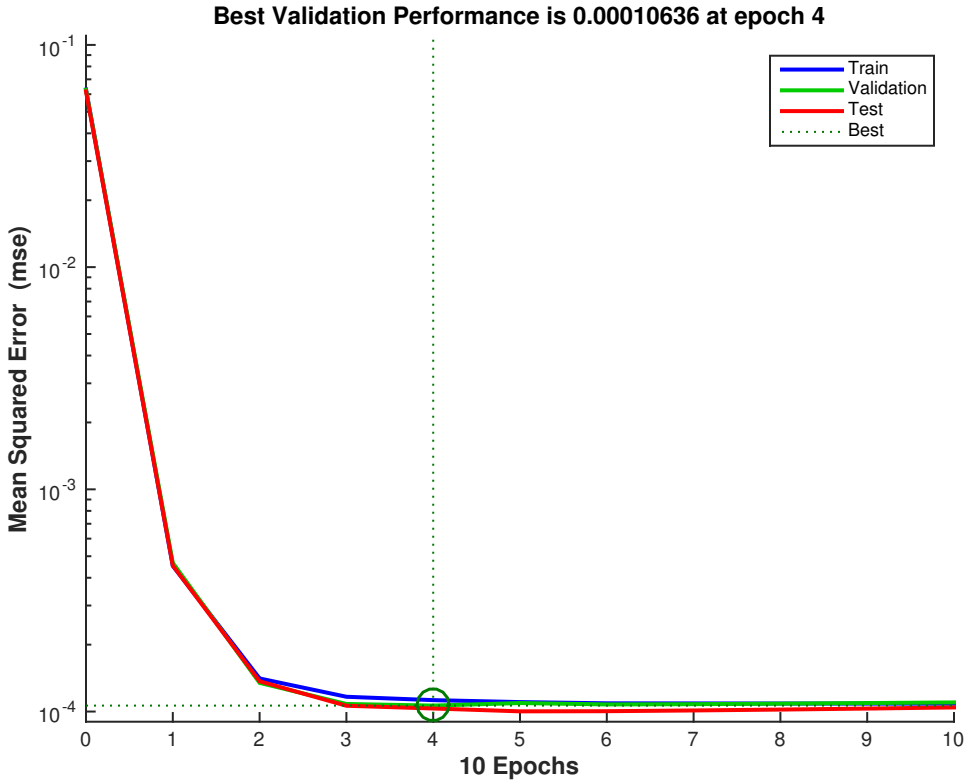


Figure 4.25: Learning curve for the neural network estimating the vibration amplitude at 20,138 Hz. Training stops when 6 subsequent iterations fail to lower the validation error.

Summarizing the model development and evaluation phase

Four models has been developed showing that the more non-linearity is taken into account, the better the model performs. Resulting performances for each model and configuration is summarized in table 4.3. Among the candidates, the neural nets provides the better fit. Both hidden layer configurations showed nearly equal performance, but the 20 neuron hidden layer performed slightly better at generalization and is therefore chosen as the final model.

Table 4.3: Considering each input individually to indicate degree of correlation.

	Model type				
MSE for training/testing:	Linear	2^{nd} degree polynomial	3^{rd} degree Polynomial	Neural Net (10 neuron)	Neural Net (20 neuron)
Average MSE learning	0.00855	0.00713	0.00617	0.00562	0.00554
Average MSE testing	0.00856	0.00714	0.00618	0.00564	0.00557

4.4.3 Residual analysis

Now, having established the final model, the residuals can readily be calculated. Figure 4.26 shows an arbitrarily chosen record with its true response plotted together with the estimated output from the neural network model. The difference between the two are calculated for all records, resulting in the *residual matrix*.

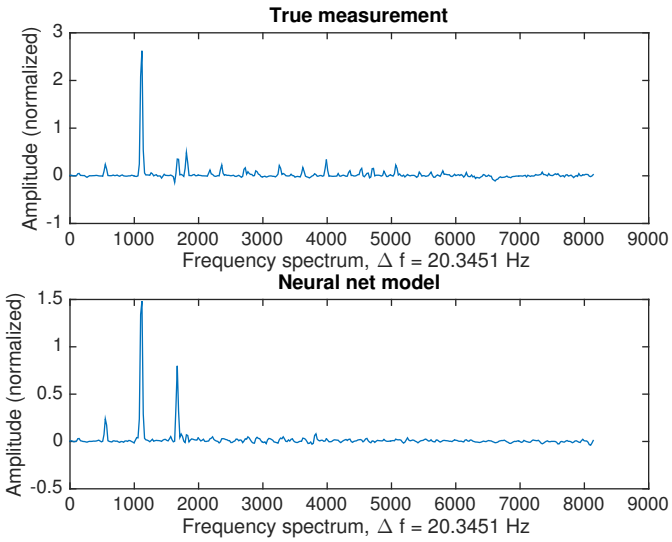


Figure 4.26: True amplitudes (above) and the amplitudes estimated by the neural net model (below).

Hopefully, the residual matrix allows for finding answers to the questions below, all of which are being presented and discussed next.

- Has the effect from power been successfully removed?
- Can anomalous behavior be identified?
- If yes, are the development consistent with a given cluster potentially representing a specific fault?
- If the answer to one or several of the above question is no: What can be plausible explanations?

Chapter 5

Results and Discussion

Followed by presenting key findings from the residual analysis, the chapter has a dedicated section discussing the goodness of the approach.

5.1 Presenting Findings from the Residual Analysis

The first objective was to separate vibrations caused by rated power and wind speed from those being a result of wear and tear. As rated power and wind speed are correlated, they will simply be referred to as *power* in the following. Figure 5.1 shows that indeed the effect from power has been successfully removed. The reoccurring pattern formed in each of the power-bins shows that it is no longer possible to discriminate vibration response according to power. The next step is then to investigate whether the residuals are strictly representing degradation or if other phenomena may have an impact.

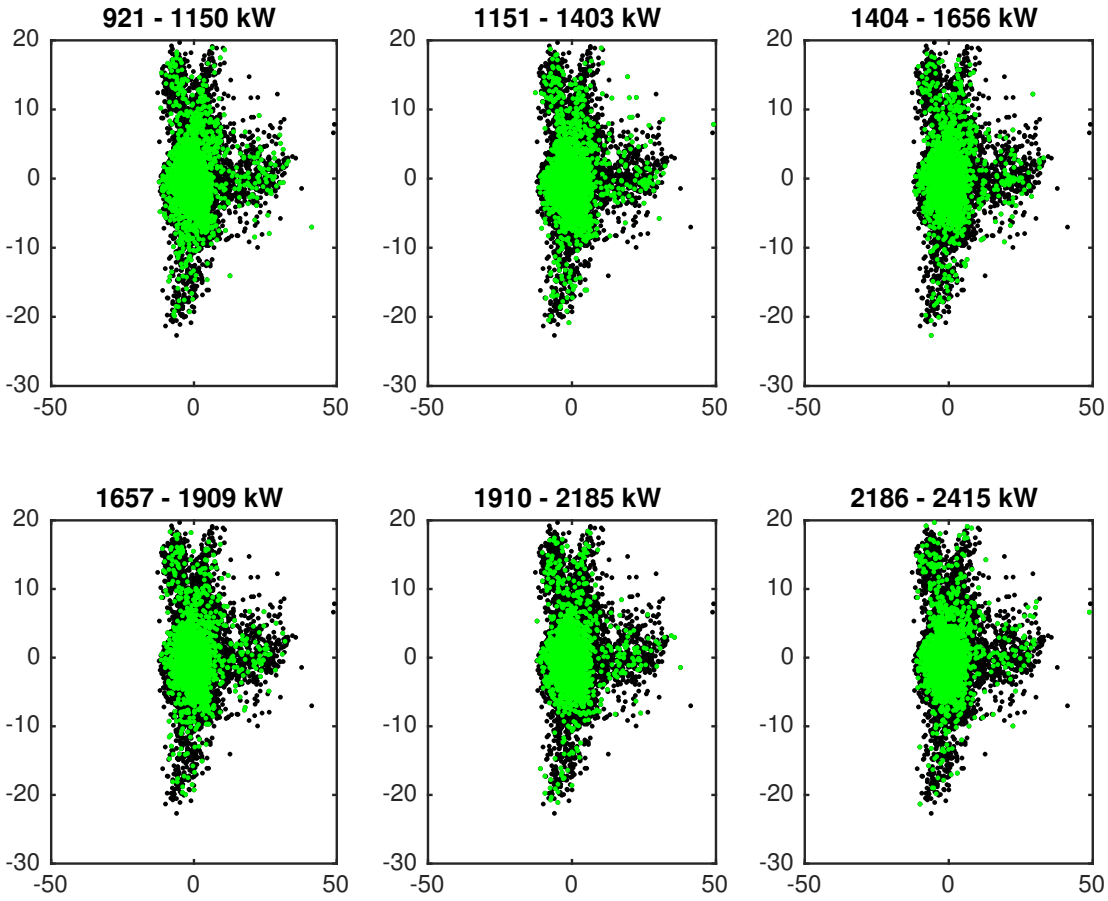
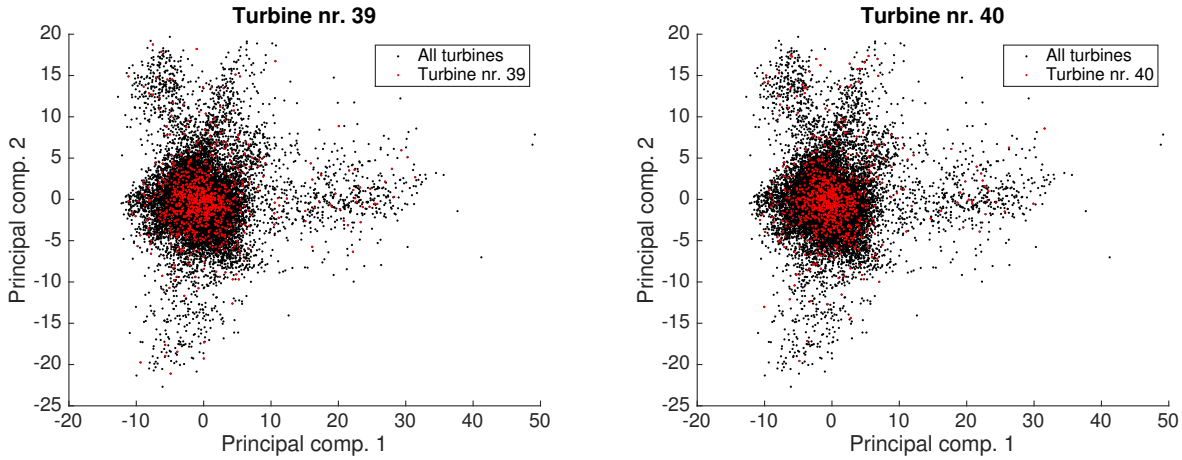


Figure 5.1: Turbine responses per bin when considering the residual. Dependence between power and vibration response can no longer be observed.

When evaluating the residuals according to turbines, at least four distinct clusters arise. Among the 20 turbines under consideration, all turbines are sporadically present in each of these clusters. Turbine 39 and 40 have been selected for illustration and are shown in figure 5.2. They are arbitrarily selected since any choice would be representative for the complete set of turbines. Recalling the simplified use-case (section 4.1.2), it was precisely the type of patterns shown in figure 5.2 we could hope for, as they potentially are representing faulted states. To evaluate whether the clusters indeed are manifests of faulted states, the transitions between them is plotted as a function of time.

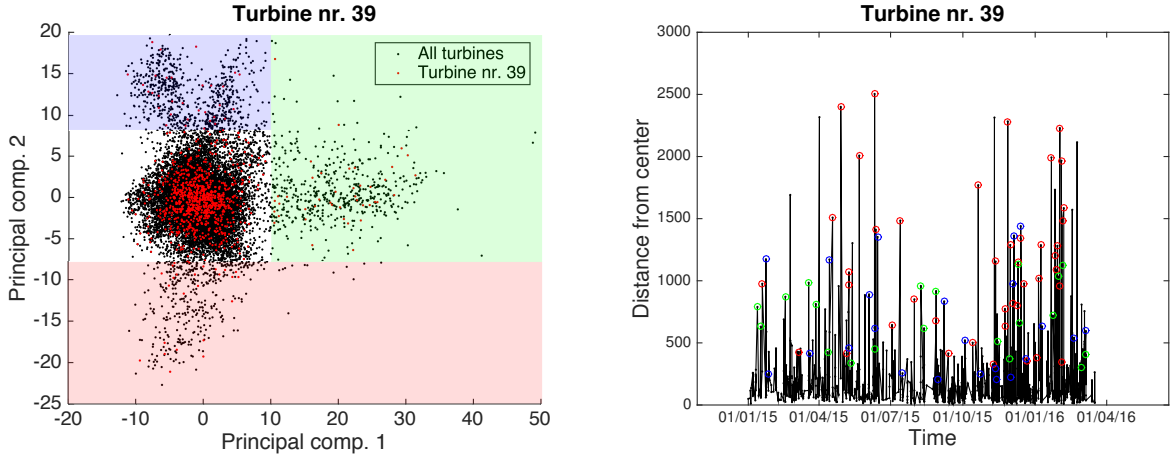


(a) Residual for T39 (red) plotted onto entire residual from T21-T40 (black)

(b) Analogously for Turbine 40

Figure 5.2: Figure show 4-5 distinct clusters and indicates the fraction of time two representative turbines are present in them.

The above clusters can in other words be considered *necessary but insufficient* prerequisites for degradation tendencies to be observed. Thus, in the same manner as outlined in 4.1.2, figure 5.3 shows how presence in the various states varies according to time.



(a) Three unlabeled zones outside "normal"

(b) Relating presence in each state to time

Figure 5.3: The rapid transitions between different clusters is not consistent with the degradation process.

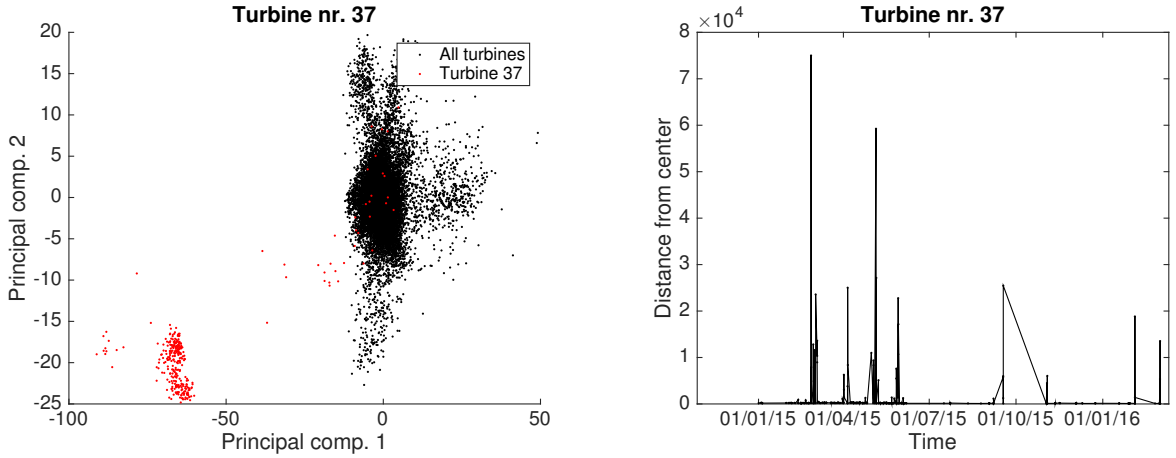
'Distance from center' in the above figure refers to the distance from the center of the major cluster that is assumed to represent normal behavior (white background). Furthermore a coarse division is made by assigning each of the minor clusters with a color. The color-wise convention

is translated to the time-relating plot to allow for identifying characteristic behavior associated with a particular cluster. If the clusters where representing faults, sustained presence in one of the fields would be expected. This turn out not to be the case. As shown for turbine 39, rapid transitions between states occur in an seemingly arbitrary manner. Nor is the overall distance from the center demonstrating increasing behavior. Similar plots were made also for the other turbines. None of them where showing behavior that is consistent with evolving faults.

Although turbine 39 is representative for the other turbines, this do not encompass turbine 37 which was discarded in the filtering-stage for showing deviating behavior.

What happened to turbine 37?

While the average number of records for the other turbines is between 1600 and 1700, T37 had only 815 records. As shown in figure 5.4 there are several periods where no records are collected. As from July 2015 records are stored merely sporadically. Plausible interpretations are take-outs, repairs, or other incidents disrupting the production regularity. Moreover, corrupted sensors could be an explanation.



(a) Turbine 37 response cluster

(b) Distance from center

Figure 5.4: The rapid transitions between different clusters is not consistent with the degradation process.

However, the fact that periods with missing records repeatedly are occurring after peaks in distance can suggest that the turbine experienced some fault, or even failure. The definite answer to what happened to turbine 37 can not be given without consulting the fault log.

5.2 Discussion

One explanation to why the results are not showing behavior that is consistent with degradation could simply be that none of the gearboxes did suffer from degradation. Statistics from GCube (2016) shows that each year, one out of 145 wind gearboxes experience a failure. Based on the experience of Salomonsen (2016), the gearbox typically is changed approximately once during its lifetime of 20 years. Furthermore, it is typically a matter of years before noticeable wear manifests (Salomonsen, 2016). Consequently, it is a fully viable option that evolving faults in fact has not been present. This is particularly true as the 14,5 month time horizon (01.01.2015-16.03.2016) might have been too small for degradation to manifest, especially if maintenance or replacements had been made shortly prior to the start of the considered time period. However, if degradation/evolving faults indeed has been present in one or several of the gearboxes, the approach used in this thesis has not been able to detect it.

Continuing the discussion assuming that the planetary stage of the gearbox indeed did experience evolving faults during the considered time period, the task becomes to identify possible reasons why the approach failed to identify it. This task can be approached systematically by considering each of the steps having been made: 1) Pre-processing/filtering; 2) model development and 3) residual analysis.

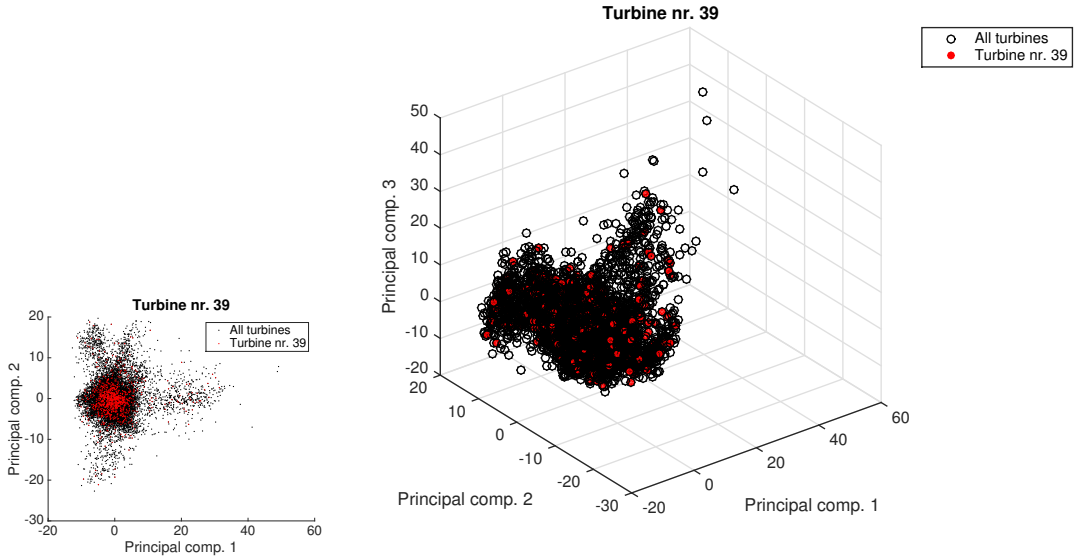
During the pre-processing/filtering phase, turbine offsets was corrected for by means of median centering. This relies on the assumption that the turbines are comparable. Intuitively this makes sense, because the turbines are all the same make, located at more or less the same coordinate, and are commissioned at the same time. Anyhow, there will always be sources to variability. Salomonsen (2016) points to differences within the production stage, as well as during operation and maintenance. Isolated, these might be just tiny contributions. However, when added together, they could make an actual difference. Consequently, the question is not whether the assumption is wrong, but rather *how* wrong it is.

Moving to the modeling phase, the generalization error is clearly decreasing as the more non-linearity is taken into account. This shows that the relation between operating conditions and vibration response indeed has a non-linear nature, therefor choosing the 20 neuron ANN as the final model. However, the model still suffer from noticeable error, especially around the

peak values. A possible way to remedy this problem is demonstrated by Porotsky and Bluvband (2012) who describe accumulated degradation of bearings in terms of accumulated acceleration. This would make the model less susceptible to "miss" on the peak values, and could potentially reduce the overall error significantly.

Another plausible explanation is that other co-variables than rated power and wind speed yields a noticeable contribution to the vibration response. In this case, the residual will not represent only the contribution from degradation, but will be influenced by phenomena that is not taken into account during the generalization phase. The relevance of this aspect was demonstrated in the use-case showing that when an appropriate generalization model is trained with all the impacting operating conditions, the approach is fully functional. It is difficult to say to what extent rated power and wind speed is not representing the entire set of operating conditions that is influencing the vibration response. What on the other hand is safe to say, is that rater power has an impact on the resulting vibration response (ref. fig. 4.19) and that the artificial neural networks are able to remove this effect (ref. fig. 5.1). Access to further operational parameters (pressure, temperature etc.) from the SCADA-system could reveal a potential correlation with other co-variables.

Another weakness is identified during the residual analysis phase. The explained variance using two principle components is merely 11 %. Figure 5.5 seeks to demonstrate this issue by showing the response-plot for turbine 39 using both two and three principal components.



(a) Two principal comp. response plot

(b) Three principal comp. response plot.

Figure 5.5: Vibration response visualized in three dimensions (right) shows that information regarding the actual location of the responses get lost when reduced to two principal components (left).

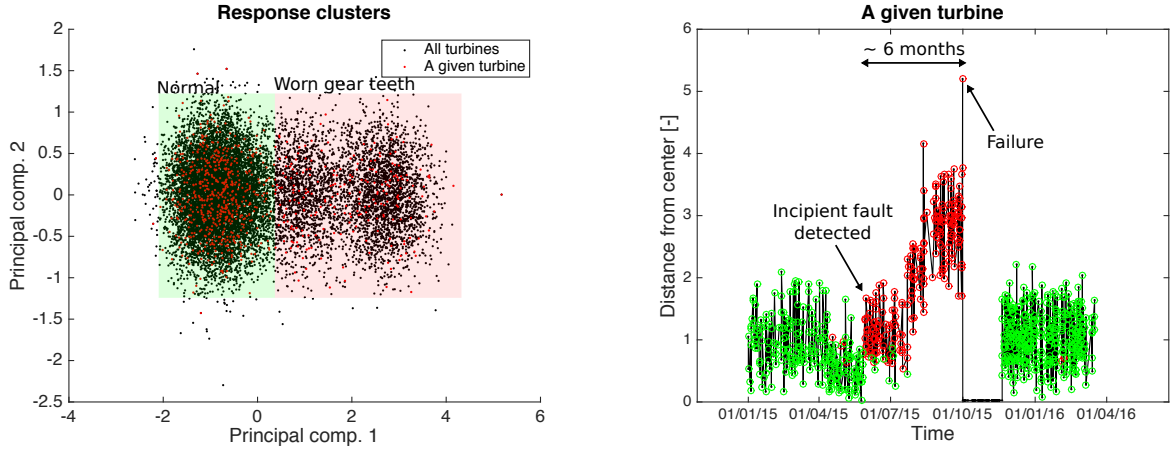
The figure shows that potentially valuable information get lost when dimensionality is reduced. In effect, this means that the clusters shown in two dimensions (left) represents significant uncertainty and are thus unfit in terms of representing system states.

This issue is not primarily a problem for the condition monitoring part, which is based on the distance from the center of the major cluster. When calculating the distance from the cluster center, 20 principal components has been used, yielding an explained variance of 52 % which is fairly adequate for this purpose. It is rather a problem in terms of *usability* that the lacking variance capture poses a problem. As significant amount of the data variability fails to be described in two and three dimensions, it might be misleading to classify the system equipment according to the observed clusters, as they might represent such a large variety of phenomena. Thus the colored labels used in figure 5.3 in the results section might have been a counterproductive representation. This is due to the fact that being member of a given color field still leaves large-grain uncertainty as the remaining expansion in the high-dimensional room is poorly explained. For the continued work it is thus recommended to consider alternative visualization techniques that are better able to capture the variance in the dataset.

5.2.1 Contextualizing the results

Although there has been identified weaknesses, they do not indicate that the approach per se has failed. The simplified use-case shows on the other hand that the approach in it self is able to reveal evolving faults efficiently. By correcting for the identified weaknesses, the proposed approach is believed to enable extraction of valuable information.

This is shown by reintroducing the gradual degradation scenario from the simplified use-case in section 4.1.2. With reference to figure 5.6 b), suppose that a visual inspections made in the beginning of June 2015 could confirm incipient wear on the gear teeth for a given turbine. This information would allow for labeling the clusters in a) as well as drawing the incipient-fault-to-failure progression timeline in b).



(a) Gradually increasing vibr. amplitude

(b) Time-dependent increase in distance

Figure 5.6: a) shows labeled clusters for vibration response b) shows that the residuals are representing fault evolution by relating the transitions to time (modified version of figure from section 4.1.2.)

Moreover, having a statistically sufficient number of such timelines would allow for building probability distributions of the fault-to-failure periods. This would naturally be done for as many failure mechanisms as there is information to describe.

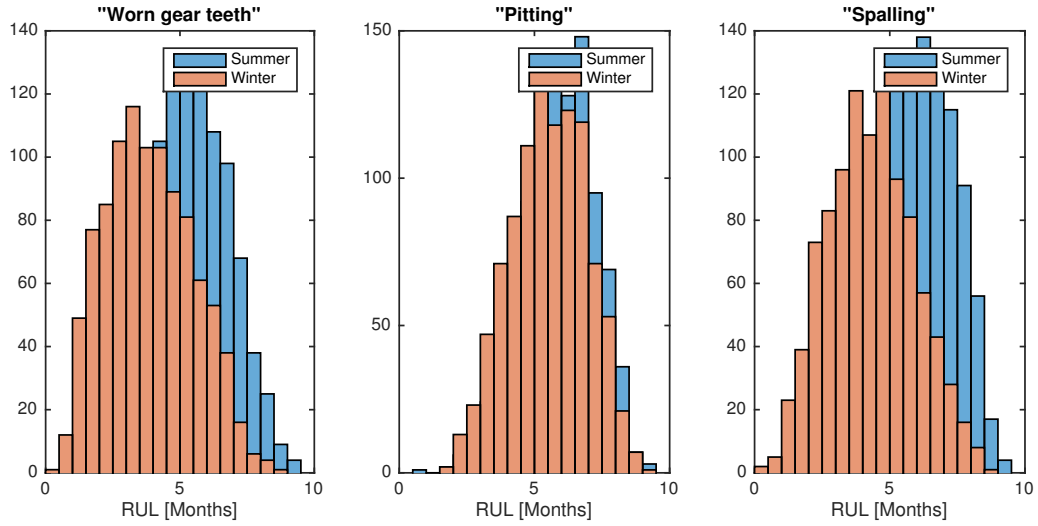


Figure 5.7: Demonstration plot showing aggregated RUL estimates for common failure mechanisms. Filtering according to season indicate the importance of considering historical load and expected future load when managing fault-failure periods for variable load machines.

The probability distributions showed above would in many cases be the enabling element of the prognostic block in the PHM-cycle - at least for "steady-state" systems. If the incipient fault is successfully detected and isolated, they would provide an estimate of the remaining useful life of the component under consideration. For wind turbines on the other hand, the irregular load give rise to additional considerations. First, the fault-to-failure periods should be filtered according to what load the machine has been experiencing, and secondly, a sufficiently accurate weather forecasting model should be in place to select the appropriate RUL-estimation. A very simple refinement is illustrated above where it is distinguished between summer and winter, as the latter tend to come with harsher weathering.

To carry out this estimation, large amounts of data will be required. Large amount of the *correct* data. Although this has been a lacking ingredient for this project, it has been proposed an approach to condition monitoring which could be fully functional when corrected for present flaws. When corrected for current flaws, the suggested approach serves as a "first step" on which access to data would determine whether or not the prognostic part of the PHM-cycle could successfully be carried out.

5.3 Summarizing the Discussion

The fact that no degradation pattern is identified can either be related to the fact that no degradation has been present during the considered time period. If assuming that the planetary stage of the gearbox indeed has been subjected to degradation, the proposed approach has arguably failed due to the following:

Median centering: Median-centering relies on assumption that the turbines are comparable units - which might be wrong.

Input-variables: Rated power and wind speed might not be sufficient co-variables for explaining the vibration.

Generalization error: The effect from missing the peak vibration amplitudes is considerable.

Dimensionality reduction: Potentially valuable information is lost when reducing the dimensionality. Employing techniques maintaining more data-variability is suggested.

Relying on access to the appropriate data it is shown that the approach can serve as a first stage in the prognostic and health management cycle by allowing for aggregating incipient fault to failure progression timelines.

5.4 Further Work

The first priority should be to get a hold of fault logs. By allowing for verification of the work already done, fault logs would guide where to put the focus when adjusting for the identified weaknesses. Access to SCADA-data should also be pursued as it could reveal potentially important co-variables impacting the vibration response.

To handle the turbine offset, multi-task learning is suggested. This learning regime does not require for an a priori determined procedure that handles the turbine-specific behavior - it is learned from the data. Consequently, it would handle the potentially wrong assumption made by doing the median-centering, namely that the turbines can be considered equivalent after all. By learning from the commonality among the multiple tasks, multitask learning often makes a

better model for its main task (Baxter, 2000). As the wind turbines indeed are representing a lot of commonality, this learning regime is believed to be well-suited for this type of analysis. The main drawback is that the most multitask learning techniques requires a lot of data to work properly (Lefebvre, 2016).

To deal with the considerable errors which is identified close to the peaks, it is suggested to consider accumulated acceleration rather than incremental values. By aggregating acceleration in intervals (bins), and making sure that they are spanning over the peak area, the model gets less susceptible to leave out considerable contributions to the actual accelerations.

It is also advised it to consider alternative techniques for visualization that enables better capture of the data variability. This would provide better usability in terms of being able to more correctly label the observed clusters.

The proposed list measures is not intended to be followed slavishly. For example, by aggregating the acceleration, the number of dimensions get reduced at the same time. Certainly this will make the PCA better able to describe the variability in the data. Perhaps to the extent where there no longer is a need to explore alternative visualization techniques. Likewise, the identification of other important co-variables impacting the vibration response might alone make for a sufficiently accurate generalization performance which in turn can rule out the need for one or several of the other suggested measures. Therefore, rather than as isolated measures for improvement, they should be considered in an intertwined manner.

Chapter 6

Conclusions

Relying on accelerometric datasets, an approach for condition monitoring of components in wind turbines has been developed and applied specifically to the planetary stage of the gearbox. Intending to be left only with the vibration response that is caused by degradation, an artificial neural network model was employed to estimate the part of the vibration that is caused by rated power and wind speed. The residuals are not influenced by rated power. This means that the model succeeded in reproducing the vibration response induced by power.

However, rated power and wind speed are possibly not covering the full range of co-variables that influences the vibration response. This can be argued because no consistency with degradation was identified during the residual analysis. Another explanation would simply be the fact that none of the gearboxes did experience evolving faults during the considered time-period.

Assuming that one or several of the gearboxes *did* suffer from degradation, the following identified weaknesses summarize possible explanations for why degradation was not detected: 1) The model is trained with an inadequate set of co-variables impacting the vibration response; 2) correcting the turbine offsets by median centering assumes that the turbines can be treated as comparable units - an assumption that evidently is not fully true; 3) insufficient generalization performance - especially with respect to being vulnerable of missing the peak values.

Notwithstanding the aforesaid, the simplified use-case shows that the approach in it self effectively detects and monitors anomalous behavior. When successfully adjusted for the aforementioned aspects, the proposed approach is believed to provide effective condition monitoring for components in a wind turbine.

Appendix A

Acronyms

ANN Artificial Neural Network

CBM Condition based maintenance

CM Condition monitoring

FFT Fast fourier transform

GA Genetic algorithm

GHG Greenhouse gas

HS Highspeed shaft

IMS Intermediate shaft

LCOE Levelized cost of electricity

MSE Mean squared error

O&M Operation and maintenance

OBM Opportunity based maintenance

OWF Offshore wind farm

PCA Principal component analysis

PHM Prognostics and health management

Planet Planetary stage of the gearbox

PO Partial Objective

RPM Revolutions per minute

RTF Run to failure

RUL Remaining useful life

SCADA Supervisory control and data aqcuisition

SVM Support vector machine

VA Vibration analysis

Appendix B

Matlab code

The appended MATLAB code are organized according to chapter.

B.1 MATLAB-code from Chapter 2 - Theoretical Framework

B.1.1 PredAccuracy.m

The value of accurate predictive information

```
1
2  clear all
3  clc
4  %% PARAMETER SETUP:
5  % cost of maintenance
6  k = 50;
7  % extra cost for preventive maintenance
8  c = 50;
9  % lifetime mean
10 m = 375;
11 % true life dist. SD
12 vdist = 50;
13 vdist2 = [2,5,10,20,50];
14 % SD in CMS (predictive output)
```

```
v = [2,5,10,20,50];
% Maintenance interaction +/-
T = [-50:0.1:50];
% Corrective
c_c = (k+c);
% Ideal
c_i = k;

%% Lifetime simulations

c_cm =[];
outputMat=[];
for x=1:length(vdist2)
for j = 1:40000
    t_hat = normrnd(m,vdist2(x),1,length(T)); % Estimated life times

for i = 1:length(t_hat)
    if t_hat(i)+T(i) > m
        c_cm(j,i) = c_c/m;
    elseif t_hat(i)+T(i) == m
        c_cm(j,i) = c_i;
    else
        c_cm(j,i) = c_i/(t_hat(i)+T(i));
    end
end

end
outputVec = mean(c_cm(:,1:length(t_hat)));

outputMat(x,:) = outputVec;
end


%% Plotting results:

figure
```

```
51
52 line([-50 50],[c_c/m c_c/m],'Color','[1 0 0]'); % Corrective
53 hold on
54 line([-50 50],[c_i/m c_i/m],'Color','[0 1 0]'); % Ideal
55 line([-50 50],[0.175 0.175],'Color','[0 0 0]','LineStyle','--');% Age based
56
57
58 plot(T(1):1:T(end),outputMat(1,1:10:end),'Color','[0,0.313725,0.6196078431]')
59 plot(T(1):1:T(end),outputMat(2,1:10:end),'Color','[0,0.313725,0.6196078431]')
60 plot(T(1):1:T(end),outputMat(3,1:10:end),'Color','[0,0.313725,0.6196078431]')
61 plot(T(1):1:T(end),outputMat(4,1:10:end),'Color','[0,0.313725,0.6196078431]')
62 plot(T(1):1:T(end),outputMat(5,1:10:end),'Color','[0,0.313725,0.6196078431]')
63
64 ylim([0.12 0.3])
65 xlabel('\Delta T') % x-axis label
66 ylabel('Average cost per time unit') % y-axis label
67 legend('Corrective','Ideal','Age based','Condition-based (\sigma=2,5,10,20,50)',...
68     'Location','northeast')
69 set(gca, 'FontSize',12)
```

B.1.2 FFT_Demo.m

Demonstrating the interest of the fast Fourier transform

```
1 %% FFT demonstration plot
2
3 Fs=1000; % Sampling frequency
4 Ts=1/Fs; % Sampling period or time step
5 dt=0:Ts:1-Ts; % Signal duration
6
7 f1=5;
8 f2=10;
9 f3=25;
10
11 y1=10*sin(2*pi*f1*dt);
```

```
y2=10*sin(2*pi*f2*dt);
y3=10*sin(2*pi*f3*dt);
yj=y1+y2+y3;
ym=y1.*y2;

nfft=length(yj); % length of time domain signal
nfft2 = 2^nextpow2(nfft); %length of signal in power of 2

ff=fft(yj,nfft2);
fff=ff(1:nfft2/2);
fff=fff/max(fff); % (Normalize)
xfft=Fs*(0:nfft2/2-1)/nfft2;


figure(1);
subplot(5,1,1)
plot(dt,y1,'r')
title('Sinusoid 1: 5 Hz')
xlabel('Time [s]')
ylabel('Ampl.')

subplot(5,1,2)
plot(dt,y2,'r')
title('Sinusoid 2: 10 Hz')
xlabel('Time [s]')
ylabel('Ampl.')

subplot(5,1,3)
plot(dt,y3,'r')
title('Sinusoid 3: 25 Hz')
xlabel('Time [s]')
ylabel('Ampl.')

subplot(5,1,4)
plot(dt,yj,'r')
title('Sinusoid 1 + Sinusoid 2 + Sinusoid 3 ')
```

```
xlabel('Time [s]')
ylabel('Ampl.')

subplot(5,1,5)
plot(xfft,abs(fff),'r');hold on
xlabel('Frequency[Hz]')
ylabel('(normalized)')
title('Frequency domain')
```

B.2 MATLAB-code from Chapter 4 - Methodology and Model Construction

B.2.1 UseCase.m

Simplified use-case where relation between input and output is linear

```
%% LOAD DATA
clear;
clc;
close all;
fprintf('LOAD DATA\n');
sTmp = load('./DataMat/Planet_FFT_0_8000');
sData = sTmp.sData;
sDataDescription = sTmp.sDataDescription;
clear sTmp;

% Obtain input-data
sDataFilt = Filtering(sData);
nbObs = size(sDataFilt.mX,1);
vPerm = randperm(nbObs);
sDataTmp = sDataFilt.mX(vPerm,:);
sDataFilt.mX = sDataTmp(1:16000,:);

% Extract Power (P), Yaw angle (Y) and wind (W)
```

```
19 P = sDataFilt.mX(:,5);
20 Y = sDataFilt.mX(:,6);
21 W = sDataFilt.mX(:,8);
22
23 %No. of frequencies to model
24 NumFreq = 2;
25
26 %% CONFIGURATION AND SETUP
27
28 % No degradation
29 if 0;
30     D = zeros(16000,1);
31     ActKey = false;
32     Failure = false;
33 end
34 % Sudden degradation
35 if 0;
36     D = [zeros(13000,1) ; ones(3000,1)];
37     ActKey = false;
38     Failure = false;
39 end
40 % Gradual degradation
41 if 1;
42     D = [zeros(2500,1); 0.2*ones(1500,1); 0.5*ones(1500,1); ...
43      0.8*ones(500,1); 1*ones(2000,1); zeros(8000,1);];
44     ActKey = true;
45     Failure = true;
46 end
47 % Inconclusive scenario (noise)
48 if 0;
49     D=[zeros(8000,1);ones(8000,1)]; % Inconclusive response
50     nbObs = length(D);
51     vPerm = randperm(nbObs);
52     D = D(vPerm);
53     ActKey = 2;
54     Failure = false;
```

```
55 end
56
57
58 % calculate vibr. from linear relation
59 e = zeros(size(sDataFilt.mX,1),NumFreq);
60
61 for i = 1:NumFreq
62 coef0 = [0 2];
63 coefP = [0.0005 0.07];
64 coefY = [5 3];
65 coefW = [5 3];
66 coefD = [3 3];
67 e(:,i) = coef0(i)+coefP(i)*P+coefY(i)*Y+coefW(i)*W+coefD(i)*D+0.5 ...
68     *randn(size(P));
69 end
70
71 %% LINEAR REGRESSION MODEL
72
73 nbObs = size(sDataFilt.mX,1);
74 ratioLearning = 0.7;
75 nLear = floor(ratioLearning*nbObs);
76 vVarMdl = [5 6 8]
77 vVarOut = [1 2]
78
79 fprintf('LINEAR MODEL\n');
80 mXLearnLin = sDataFilt.mX(1:nLear,vVarMdl);
81 mYLearnLin = e(1:nLear,vVarOut);
82 mXTestLin = sDataFilt.mX(nLear+1:end,vVarMdl);
83 mYTestLin = e(nLear+1:end,vVarOut);
84 [mXLearnLin,muLearnLin,sigmaLearnLin] = zscore(mXLearnLin);
85 mXTestLin = (mXTestLin-ones(size(mXTestLin,1),1)*muLearnLin) ...
86     ./(ones(size(mXTestLin,1),1)*sigmaLearnLin);
87 mALin = nan(size(mXLearnLin,2)+1,size(mYLearnLin,2));
88 mYLearnEstLin = nan(size(mYLearnLin));
89 mYTestEstLin = nan(size(mYTestLin));
90 for idy=1:size(mYLearnLin,2)
```

```
91      X = [mXLearnLin ones(nLear,1)];
92      y = mYLearnLin(:,idy);
93      mALin(:,idy) = inv(X'*X+eye(size(X,2))*1e-6)*X'*y;
94      mYLearnEstLin(:,idy) = [mXLearnLin ones(size(mXLearnLin,1),1)]...
95          *mALin(:,idy);
96      mYTestEstLin(:,idy) = [mXTestLin ones(size(mXTestLin,1),1)]...
97          *mALin(:,idy);
98  end
99
100 % Testing and training error (MSE)
101 vLearnLinError = mean((mYLearnLin-mYLearnEstLin).^2);
102 vTestLinError = mean((mYTestLin-mYTestEstLin).^2);
103 fprintf('Linear mdl (Learning): %f\nLinear mdl (Test): %f\n', ...
104     mean(vLearnLinError),mean(vTestLinError));
105
106
107
108 %% RESPONSE CLUSTERS USING PCA
109 ResMat = [mYLearnLin-mYLearnEstLin; mYTestLin-mYTestEstLin ];
110
111 %Include failure and downtime
112 % if Failure == 1;
113 % ResMat(8000,:) = 4 ;
114 % ResMat(8001:10000,:) = zeros;
115 % end
116
117
118 [coeff,score,latent,tsquared,explained,mu]=pca(zscore(ResMat));
119 figure;
120 plot(score(:,1),score(:,2),'.k');
121 title('Residual visualized with two principal components');
122 xlabel('Principal comp. 1')
123 ylabel('Principal comp. 2')
124 set(gca,'FontSize', 14);
125
126 figure;
```

```
127 %histogram(((ResMat(:,2)).^2)/size(ResMat,1));
128 histogram(ResMat(:,2));
129 title('Distribution of model error')
130 xlabel('Model error')
131 set(gca,'FontSize', 14);
132
133
134 %% DISTANCE FROM CENTER
135
136 mXTmpAll =  [sDataFilt.mX(:,1:11) ResMat];
137 vUniTurb = unique(mXTmpAll(:,1));
138
139 for idt=10:14%numel(vUniTurb)
140     vInd = find(mXTmpAll(:,1)==vUniTurb(idt));
141     vDate = sort(mXTmpAll(vInd,11));
142     [~,vOrd] = sort(vDate);
143     vInd = vInd(vOrd);
144     figure;
145     hold on;
146
147     plot(score(:,1),score(:,2),'.k');
148     plot(score(vInd,1),score(vInd,2),'.r');
149
150     title('Response clusters')
151     xlabel('Principal comp. 1')
152     ylabel('Principal comp. 2')
153     legend('All turbines', 'A given turbine')
154     set(gca,'FontSize', 12);
155
156     % Distance more than three dimensions
157     if 0
158     vDistTmp = sum((score(vInd,1:2)-ones(numel(vInd),1)...
159     *mean(score(:,1:2))).^2,2);
160     end
161
162     % Distance uptil three dimensions
```

```
163     if 1
164     vDistTmp = sqrt((score(vInd,1)-mean(score(:,1))).^2 ...
165         +(score(vInd,2)-mean(score(:,2))).^2);
166     end
167 
168     vDateTmp = vDate(vOrd);
169     set(gca,'FontSize', 12)
170     figure;
171     plot(vDateTmp,vDistTmp,'.k-');
172 
173     % If degradation, assign colors corresponding to clusters
174     if ActKey == 1
175     hold on;
176     plot(vDateTmp(score(vInd,1)>0.5),vDistTmp(score(vInd,1)>0.5),'or');
177     vDownTime = diff(score(:,1)) == 0;
178     score(vDownTime) = zeros;
179     valueTmp = score(vDownTime);
180     value = valueTmp(1);
181     vDistTmp(vDownTime) = zeros;
182     plot(vDateTmp(score(vInd,1)<0.5 & score(vInd,1)~=value), ...
183         vDistTmp(score(vInd,1)<0.5 & score(vInd,1)~=value),'og');
184     end
185     if ActKey == 2
186     hold on;
187     plot(vDateTmp(score(vInd,1)>0.5),vDistTmp(score(vInd,1)>0.5),'or');
188     plot(vDateTmp(score(vInd,1)<0.5),vDistTmp(score(vInd,1)<0.5),'og');
189     else
190     end
191 
192     datetick('x','dd/mm/yy','keeplimits');
193     xlabel('Time');
194     ylabel('Distance from center [-]');
195     title('A given turbine')
196     set(gca,'FontSize', 14)
197     pause(0.05)
198 end
```

```
199
200 %% IDEAL DEGRADATION DEMO
201
202 x1 = [1 73];
203 y1 = [1 2];
204 figure;
205 plot(x1,y1,'Color',[0 0.4470 0.7410])
206 hold on
207 x2 = [80 200];
208 y2 = [1 2];
209 plot(x2,y2,'Color',[0 0.4470 0.7410])
210 hold on;
211 x3 = [207 279];
212 y3 = [1 2];
213 plot(x3,y3,'Color',[0 0.4470 0.7410])
214 ylim([0 4])
215 set(gca, 'XTickLabelMode', 'manual', 'XTickLabel', ['"Repair"'],...
216     'xtick', [73 200], 'YTickLabelMode', 'manual', 'YTickLabel', [],...
217     'ytick', [1 2],'FontSize', 18);
218 xlabel('Time')
219 ylabel('Distance')
220 title('Distance from normal bevavior center')
221 grid on;
222
223 %% POSSIBLE STATE INTERPRETATION
224
225 x = [1 73 73 80 80 200 200 207 207 279 279];
226 y = [1 2 0.5 0.5 1 2 0.5 0.5 1 2 0.5 ];
227 figure;
228 plot(x,y,'Color',[0 0.4470 0.7410])
229 title('System states')
230 xlabel('Time')
231 ylim([0 4])
232 ax = gca;
233 ax.YTick = [0.5 1 2];
234 ax.YTickLabel = {'Repair', 'Normal', 'Abnormal'};
```

```
235 set(gca, 'XTickLabelMode', 'manual', 'XTickLabel', [], 'xtick',...
236     [73 200],'FontSize', 18);
237 get(gca,'ColorOrder')
238 grid on;
```

B.2.2 Filtering.m

Data filtering

```
1  function sData = Filtering(sData)
2
3  %% Years
4  MyDates = datevec(sData.mX(:,11));
5  vYear = [2015,2016];
6  vFlag = ismember(MyDates(:,1),vYear);
7  sData.mX = sData.mX(vFlag,:);
8
9  %% Months
10 MyDates = datevec(sData.mX(:,11));
11 vMonth = [1:12];
12 vFlag = ismember(MyDates(:,2),vMonth);
13 sData.mX = sData.mX(vFlag,:);
14
15 %% Power bins
16 %vBin = [920 1150 1403 1656 1909 2185 2415];
17 vBin = [1150 1403 1656 1909 2185 2415];
18 vFlag = ismember(sData.mX(:,4),vBin);
19 sData.mX = sData.mX(vFlag,:);
20
21 %% Turbines
22 %vBin = [920 1150 1403 1656 1909 2185 2415];
23 vUniTurb = unique(sData.mX(:,1));
24 %remove T 37 and T 21
25 vUniTurb(17) = [];
26 vUniTurb(1) = [];
```

```
vFlag = ismember(sData.mX(:,1),vUniTurb);
sData.mX = sData.mX(vFlag,:);
end
```

B.2.3 MainAnalysis.m

The main data-analysis

```
clear;
clc;
close all;

%% LOAD DATA
fprintf('LOAD DATA\n');
sTmp = load('./DataMat/Planet_FFT_0_8000');
% sTmp = load('./DataMat/Highspeed_FFT_0_8000')
% sTmp = load('./DataMat/Generator+bag_FFT_0_8000')
% sTmp = load('./DataMat/Generator_FFT_0_8000')
% sTmp=load('./DataMat/IMS_FFT_0_8000')
sData = sTmp.sData;
sDataDescription = sTmp.sDataDescription;
clear sTmp;

% Response entire dataset (PCA)
[coeff,score,latent,tsquared,explained,mu]=pca(zscore(sData.mX(:,12:end)));
figure;
plot(score(:,1),score(:,2),'.k');
title('Entire data set');


%% FILTER DATA
fprintf('FILTER DATA\n');
sDataFilt = Filtering(sData);
%sDataFilt = Filtering_modrun(sData); % include turbine 37
nbObs = size(sDataFilt.mX,1);
```

```
28 vPerm = randperm(nbObs);
29 sDataFilt.mX = sDataFilt.mX(vPerm,:);
30
31 %% INITIAL PLOTS FOR HYPOTHESIS EVALUATION
32 %Response-plot all turbines power bin 2-7
33 [coeff,score,latent,tsquared,explained,mu] = ...
34     pca(zscore(sDataFilt.mX(:,12:end)));
35 figure;
36 plot(score(:,1),score(:,2),'.k');
37 %Add selected turbines
38 if 1
39 hold on;
40 for i = [1 3]
41 vTurb = unique(sDataFilt.mX(:,1));
42 C = {'.c','.m','.m','.g','.c','b','k',}; % Cell array of colros.
43 vFlag =  sDataFilt.mX(:,1)==vTurb(i) ;
44 h(i)=plot(score(vFlag,1),score(vFlag,2),C{i});
45 end
46 legend([h(1), h(3)],{'T 22','T 24'},'Location','east');
47 end
48 title('921 - 2415 kW');
49 xlabel('Principal comp. 1')
50 ylabel('Principal comp. 2')
51 set(gca,'FontSize', 18);
52
53 %Response-plot each turbine separately
54 vTurb = unique(sDataFilt.mX(:,1));
55 for idTurb=1:numel(vTurb)
56     figure;
57     vFlag = sDataFilt.mX(:,1)==vTurb(idTurb);
58     plot(score(:,1),score(:,2),'.k');
59     hold on;
60     plot(score(vFlag,1),score(vFlag,2),'.r');
61     title(sprintf('Turbine nr. %d',vTurb(idTurb)));
62     set(gca,'FontSize', 18)
63     xlabel('Principal comp. 1')
```

```
64      ylabel('Principal comp. 2')
65      pause(0.25);
66  end
67
68  %Response from all turbines for each power bin
69  sDataFiltTmp = Filtering_AllBin(sData);
70  [coeffTmp,scoreTmp] = pca(zscore(sDataFiltTmp.mX(:,12:end)));
71  vBinUp = [920 1150 1403 1656 1909 2185 2415];
72  vBinLow =[0 921 1151 1404 1657 1910 2186];
73  vBin = unique(sData.mX(:,3));
74  figure;
75  for idBin=1:numel(vBin);
76      vFlagBin = sDataFiltTmp.mX(:,3)==vBin(idBin);
77      subplot(2,4,idBin);
78      plot(scoreTmp(vFlagBin,1),scoreTmp(vFlagBin,2),'.k');
79      title(sprintf('%d - %d kW %',vBinLow(idBin),vBinUp(idBin)));
80  end
81
82  %Bin-wise response selected turbines
83  vBinUp = [1150 1403 1656 1909 2185 2415];
84  vBinLow =[921 1151 1404 1657 1910 2186];
85  vBin = unique(sDataFilt.mX(:,3));
86  vTurb = unique(sDataFilt.mX(:,1));
87  C = {'.r','.m','.b','.g','.c','b','k',};
88  figure;
89  for idBin=1:numel(vBin);
90      vFlagBin = sDataFilt.mX(:,3)==vBin(idBin);
91      subplot(2,3,idBin);
92      plot(score(vFlagBin,1),score(vFlagBin,2),'.k');
93      hold on;
94      for idTurb=1:4%Show turbine 22,23,24,25
95      vFlag = sDataFilt.mX(:,3)==vBin(idBin) & ...
96          sDataFilt.mX(:,1)==vTurb(idTurb) ;
97      h(idTurb)=plot(score(vFlag,1),score(vFlag,2),C{idTurb});
98      end
99      title(sprintf('%d - %d kW %',vBinLow(idBin),vBinUp(idBin)));
```

```
100     legend([h(1), h(2),h(3),h(4)],{'T 22','T 23','T 24','T 25'},...
101         'Location','east');
102 end
103 
104 %Response-plot per power-bin
105 vBin = unique(sDataFilt.mX(:,3));
106 for idBin=1:numel(vBin)
107     figure;
108     vFlag = sDataFilt.mX(:,3)==vBin(idBin);
109     plot(score(:,1),score(:,2),'.k');
110     hold on;
111     plot(score(vFlag,1),score(vFlag,2),'.g');
112     title(sprintf('Bin nr. %d',idBin));
113     pause(0.25);
114 end
115 
116 %% MEDIAN CENTERING
117 sDataFiltCentre = sDataFilt;
118 vTurb = unique(sDataFilt.mX(:,1));
119 for idTurb=1:numel(vTurb)
120     vFlag = sDataFilt.mX(:,1)==vTurb(idTurb);
121     sDataFiltCentre.mX(vFlag,12:end) = sDataFilt.mX(vFlag,12:end) ...
122         -ones(sum(vFlag==1),1)*median(sDataFilt.mX(vFlag,12:end));
123 end
124 
125 %% POST CENTERING PLOTS
126 
127 %Bin 2-7 after  after centering
128 [coeff,score,latent,tsquared,explained,mu] = ...
129 pca(zscore(sDataFiltCentre.mX(:,12:end)));
130 sum(explained(1:5));
131 figure;
132 plot(score(:,1),score(:,2),'.k');
133 %Show turbine 22 and 24
134 if 1
135 hold on;
```

```
for i = [1 3]
vTurb = unique(sDataFilt.mX(:,1));
C = {'.c','.m','.m','.g','.c','b','k',};
vFlag =  sDataFilt.mX(:,1)==vTurb(i) ;
h(i)=plot(score(vFlag,1),score(vFlag,2),C{i});
end
legend([h(1), h(3)],{'T 22','T 24'},'Location','east');
end
set(gca,'FontSize', 18);
xlabel('Principal comp. 1')
ylabel('Principal comp.2')
title(' 921 - 2415 kW - after centering');

%Turbine-wise response after centering
for idTurb=1:numel(vTurb)
    figure;
    vFlag = sDataFiltCentre.mX(:,1)==vTurb(idTurb);
    plot(score(:,1),score(:,2),'.k');
    hold on;
    plot(score(vFlag,1),score(vFlag,2),'.r');
    set(gca,'FontSize', 18);
    xlabel('Principal comp. 1')
    ylabel('Principal comp.2')
    title(sprintf('Turbine nr. %d',vTurb(idTurb)));
end

%Bin-wise response after centering
vBin = [1150 1403 1656 1909 2185 2415];
vBinLow =[921 1151 1404 1657 1910 2186];
vBin = unique(sDataFiltCentre.mX(:,4));
for idBin=1:numel(vBin)
    vFlag = sDataFiltCentre.mX(:,4)==vBin(idBin);
    subplot(2,3,idBin)
    plot(score(:,1),score(:,2),'.k');
    hold on;
    plot(score(vFlag,1),score(vFlag,2),'.g');
```

```
172     title(sprintf('%d - %d kW %',vBinLow(idBin),vBin(idBin)));
173 end
174 
175 %% CORRELATION BETWEEN RATED POWER, WIND SPEED AND YAW ANGLE
176 vVarMdl = [5 8 6];
177 figure;
178 subplot(2,2,1);
179 plot(sDataFiltCentre.mX(:,vVarMdl(1)), ...
180     sDataFiltCentre.mX(:,vVarMdl(2)),'.k');
181 set(gca,'FontSize', 11);
182 xlabel(sDataFiltCentre.cX{vVarMdl(1)});
183 ylabel(sDataFiltCentre.cX{vVarMdl(2)});
184 grid on;
185 subplot(2,2,2);
186 plot(sDataFiltCentre.mX(:,vVarMdl(3)), ...
187     sDataFiltCentre.mX(:,vVarMdl(2)),'.k');
188 set(gca,'FontSize', 11);
189 xlabel(sDataFiltCentre.cX{vVarMdl(3)});
190 ylabel(sDataFiltCentre.cX{vVarMdl(2)});
191 grid on;
192 subplot(2,2,3);
193 plot(sDataFiltCentre.mX(:,vVarMdl(1)), ...
194     sDataFiltCentre.mX(:,vVarMdl(3)),'.k');
195 set(gca,'FontSize', 11);
196 xlabel(sDataFiltCentre.cX{vVarMdl(1)});
197 ylabel(sDataFiltCentre.cX{vVarMdl(3)});
198 grid on;
199 
200 %% MODEL INPUT AND LEARNING RATIO FOR CROSS-VALIDATION
201 ratioLearning = 0.7;
202 nLear = floor(ratioLearning*nbObs);
203 vVarMdl = [5 8];
204 vVarOut = floor(linspace(12,412,401));
205 sDataFilt = sDataFiltCentre;
206 %% LINEAR MODEL
207 fprintf('LINEAR MODEL\n');
```

```
208
209 % Set-up for cross validation
210 k = 10;
211 vLearnLinError = zeros((k),401);
212 vTestLinError = zeros((k),401);
213 parts = floor(linspace(1,size(sDataFilt.mX,1),(k+1)));
214 parts(1)=0;
215 Rows = (1:size(sDataFilt.mX,1))';
216
217 % Train linear model with 10-fold cross validation
218 for CvL = 1:k
219     testFlag = ismember(Rows,Rows((parts(CvL)+1):parts(CvL+1)));
220     mXLearnLin = sDataFilt.mX(~testFlag,vVarMdl);
221     % randTmp = randperm(size(mXLearnLin,1)); % radom input
222     % mXLearnLin = mXLearnLin(randTmp,:); % random input
223     mYLearnLin = sDataFilt.mX(~testFlag,vVarOut);
224     mXTestLin = sDataFilt.mX(testFlag,vVarMdl);
225     mYTestLin = sDataFilt.mX(testFlag,vVarOut);
226     [mXLearnLin,muLearnLin,sigmaLearnLin] = zscore(mXLearnLin);
227     mXTestLin = (mXTestLin-ones(size(mXTestLin,1),1)*muLearnLin) ...
228         ./(ones(size(mXTestLin,1),1)*sigmaLearnLin);
229     mALin = nan(size(mXLearnLin,2)+1,size(mYLearnLin,2));
230     mYLearnEstLin = nan(size(mYLearnLin));
231     mYTestEstLin = nan(size(mYTestLin));
232         for idy=1:size(mYLearnLin,2)
233             X = [mXLearnLin ones(size(mXLearnLin,1),1)];
234             y = mYLearnLin(:,idy);
235             mALin(:,idy) = inv(X'*X+eye(size(X,2))*1e-6)*X'*y;
236             mYLearnEstLin(:,idy)=[mXLearnLin ones(size(mXLearnLin,1),1)]...
237                 *mALin(:,idy);
238             mYTestEstLin(:,idy) = [mXTestLin ones(size(mXTestLin,1),1)]...
239                 *mALin(:,idy);
240         end
241 % Store MSE for each fold
242 vLearnLinError(CvL,:) = mean((mYLearnLin-mYLearnEstLin).^2); %MSE training
243 vTestLinError(CvL,:) = mean((mYTestLin-mYTestEstLin).^2); % MSE testing
```

```
244 end
245
246 %Average error within each fold
247 vLearnLinError = mean(vLearnLinError);
248 vTestLinError = mean(vTestLinError);
249
250 %Print resulting error to screen
251 fprintf('Linear mdl (Learning): %f\nLinear mdl (Test): %f\n', ...
252     mean(vLearnLinError'),mean(vTestLinError'));
253
254 %Plot error per freq (MSE).
255 figure();
256 hold on;
257 plot(sDataDescription.vX,vLearnLinError,'r');
258 plot(sDataDescription.vX,vTestLinError,'g');
259 xlabel('Frequencies [Hz]');
260 ylabel('MSE');
261 title('Linear Model');
262 grid on;
263 legend('learning', 'testing')
264 set(gca,'FontSize', 14);
265
266 %% POLYNOMIAL MODEL SECOND ORDER
267 fprintf('POLY MODEL (2)\n');
268
269 % Set-up for cross validation
270 k = 10;
271 vLearnPolyError = zeros((k),401);
272 vTestPolyError = zeros((k),401);
273 parts = floor(linspace(1,size(sDataFilt.mX,1),(k+1)));
274 parts(1)=0;
275 Rows = (1:size(sDataFilt.mX,1))';
276
277 % Train 2nd order poly with 10-fold cross validation
278 for CvL = 1:k
279     testFlag = ismember(Rows,Rows((parts(CvL)+1):parts(CvL+1)));
```

```
280     mXLearnPoly = sDataFilt.mX(~testFlag,vVarMdl);
281     mXLearnPoly = [mXLearnPoly.^2 mXLearnPoly];
282     %randTmp = randperm(size(mXLearnPoly,1)); % randomize order
283     %mXLearnPoly = mXLearnPoly(randTmp,:); % randomize order
284     mYLearnPoly = sDataFilt.mX(~testFlag,vVarOut);
285     mXTestPoly = sDataFilt.mX(testFlag,vVarMdl);
286     mXTestPoly = [mXTestPoly.^2 mXTestPoly];
287     mYTestPoly = sDataFilt.mX(testFlag,vVarOut);
288     [mXLearnPoly,muLearnPoly,sigmaLearnPoly] = zscore(mXLearnPoly);
289     mXTestPoly = (mXTestPoly-ones(size(mXTestPoly,1),1)*muLearnPoly) ...
290         ./(ones(size(mXTestPoly,1),1)*sigmaLearnPoly);
291     mAPoly = nan(size(mXLearnPoly,2)+1,size(mYLearnPoly,2));
292     mYLearnEstPoly = nan(size(mYLearnPoly));
293     mYTestEstPoly = nan(size(mYTestPoly));
294     for idy=1:size(mYLearnPoly,2)
295         X = [mXLearnPoly ones(size(mXLearnPoly,1),1)];
296         y = mYLearnPoly(:,idy);
297         mAPoly(:,idy) = inv(X'*X+eye(size(X,2))*1e-6)*X'*y;
298         mYLearnEstPoly(:,idy)=[mXLearnPoly ones(size(mXLearnPoly,1),1)]...
299             *mAPoly(:,idy);
300         mYTestEstPoly(:,idy) = [mXTestPoly ones(size(mXTestPoly,1),1)]...
301             *mAPoly(:,idy);
302         %keyboard;
303     end
304 %Store error of each fold
305 vLearnPolyError(CvL,:) = mean((mYLearnPoly-mYLearnEstPoly).^2); %MSE train
306 vTestPolyError(CvL,:) = mean((mYTestPoly-mYTestEstPoly).^2); %MSE test
307 end
308
309 %Average error over all folds
310  vLearnPolyError = mean(vLearnPolyError);
311  vTestPolyError = mean(vTestPolyError);
312
313 %Print error to screen
314 fprintf('Poly mdl (Learning): %f\nPoly mdl (Test): %f\n', ...
315     mean(vLearnPolyError'),mean(vTestPolyError'));
```

```
316
317 %Plot error per frequency increment
318 figure();
319 hold on;
320 plot(sDataDescription.vX,vLearnPolyError,'r');
321 plot(sDataDescription.vX,vTestPolyError,'g');
322 xlabel('Frequencies [Hz]');
323 ylabel('MSE');
324 title('2^{nd} deg. Polynomial Model');
325 grid on;
326 legend('learning', 'testing')
327 set(gca,'FontSize', 18);
328
329 %% POLYNOMIAL MODEL THIRD ORDER
330 fprintf('POLY MODEL (3)\n');
331
332 % Set-up for cross validation
333 k = 10;
334 %vLearnPolyRMSEN = zeros((k),401);
335 %vTestPolyRMSEN = zeros((k),401);
336 vLearnPolyError = zeros((k),401);
337 vTestPolyError = zeros((k),401);
338 parts = floor(linspace(1,size(sDataFilt.mX,1),(k+1)));
339 parts(1)=0;
340 Rows = (1:size(sDataFilt.mX,1))';
341
342 % Train 3rd order. poly with 10-fold cross validation
343 for CvL = 1:k
344     testFlag = ismember(Rows,Rows((parts(CvL)+1):parts(CvL+1)));
345     mXLearnPoly = sDataFilt.mX(~testFlag,vVarMdl);
346     mXLearnPoly = [mXLearnPoly.^3 mXLearnPoly.^2 mXLearnPoly];
347 %   randTmp = randperm(size(mXLearnPoly,1)); % randomize order
348 %   mXLearnPoly = mXLearnPoly(randTmp,:); % randomize order
349     mYLearnPoly = sDataFilt.mX(~testFlag,vVarOut);
350     mXTestPoly = sDataFilt.mX(testFlag,vVarMdl);
351     mXTestPoly = [mXTestPoly.^3 mXTestPoly.^2 mXTestPoly];
```

```
352     mYTestPoly = sDataFilt.mX(testFlag,vVarOut);
353     [mXLearnPoly,muLearnPoly,sigmaLearnPoly] = zscore(mXLearnPoly);
354     mXTestPoly = (mXTestPoly-ones(size(mXTestPoly,1),1)*muLearnPoly) ...
355         ./(ones(size(mXTestPoly,1),1)*sigmaLearnPoly);
356     mAPoly = nan(size(mXLearnPoly,2)+1,size(mYLearnPoly,2));
357     mYLearnEstPoly = nan(size(mYLearnPoly));
358     mYTestEstPoly = nan(size(mYTestPoly));
359     for idy=1:size(mYLearnPoly,2)
360         X = [mXLearnPoly ones(size(mXLearnPoly,1),1)];
361         y = mYLearnPoly(:,idy);
362         mAPoly(:,idy) = inv(X'*X+eye(size(X,2))*1e-6)*X'*y;
363         mYLearnEstPoly(:,idy)=[mXLearnPoly ones(size(mXLearnPoly,1),1)]...
364             *mAPoly(:,idy);
365         mYTestEstPoly(:,idy) = [mXTestPoly ones(size(mXTestPoly,1),1)]...
366             *mAPoly(:,idy);
367     end
368 % Store error of each fold
369 vLearnPolyError(CvL,:) = mean((mYLearnPoly-mYLearnEstPoly).^2);%MSE train
370 vTestPolyError(CvL,:) = mean((mYTestPoly-mYTestEstPoly).^2); %MSE test
371 end
372
373 %Average error over all folds
374  vLearnPolyError = mean(vLearnPolyError);
375  vTestPolyError = mean(vTestPolyError);
376
377 %Print error to screen
378 fprintf('Poly mdl (Learning): %f\nPoly mdl (Test): %f\n', ...
379     mean(vLearnPolyError'),mean(vTestPolyError'));
380
381 %Plot error per frequency increment
382 figure();
383 hold on;
384 plot(sDataDescription.vX,vLearnPolyError,'r');
385 plot(sDataDescription.vX,vTestPolyError,'g');
386 xlabel('Frequencies [Hz]');
387 ylabel('MSE');
```

```
388 title('3^{rd} deg. Polynomial Model');
389 grid on;
390 legend('learning', 'testing')
391 set(gca,'FontSize', 14);
392
393 %Plot residual for third order polynomial
394 mXTmp = [mYTestPoly-mYTestEstPoly ; mYLearnPoly-mYLearnEstPoly];
395 [coeff,score,latent,tsquared,explained,mu]=pca(zscore(mXTmp));
396 figure;
397 plot(score(:,1),score(:,2),'.k');
398 title('All data set Residual Poly (2) Mdl');
399 xlabel('Principle comp. 1')
400 ylabel('Principle comp. 2')
401 set(gca,'FontSize', 18);
402
403 %% CREATE ARTIFICIAL NEURAL NETWORKS
404 %Train artificial neural networks
405 mXLearnNNet = sDataFilt.mX(1:nLear,vVarMdl);
406 mYLearnNNet = sDataFilt.mX(1:nLear,vVarOut);
407 mXTestNNet = sDataFilt.mX(nLear+1:end,vVarMdl);
408 mYTestNNet = sDataFilt.mX(nLear+1:end,vVarOut);
409 [mXLearnNNet,muLearnNNet,sigmaLearnNNet] = zscore(mXLearnNNet);
410 mXTestNNet = (mXTestNNet-ones(size(mXTestNNet,1),1)*muLearnNNet) ...
411     ./(ones(size(mXTestNNet,1),1)*sigmaLearnNNet);
412 mANNet = nan(size(mXLearnNNet,2)+1,size(mYLearnNNet,2));
413 mYLearnEstNNet = nan(size(mYLearnNNet));
414 mYTestEstNNet = nan(size(mYTestNNet));
415 t0 = tic;
416 for idy=1:1%size(mYLearnNNet,2)
417     t00 = tic;
418     net = fitnet(20,'trainlm');
419     [net,tr] = train(net,mXLearnNNet',mYLearnNNet(:,idy)');
420     cNet1{idy} = net;
421     cTr1{idy} = tr;
422     mYLearnEstNNet(:,idy) = net(mXLearnNNet')';
423     mYTestEstNNet(:,idy) = net(mXTestNNet')';
```

```
424     fprintf('%d/%d (%f)\n',idy,size(mYLearnNNet,2),toc(t00));
425     %keyboard;
426 end
427 fprintf('TOTAL %f\n',toc(t0));
428 
429 %Store mean squared error
430 vLearnNNetError = mean((mYLearnNNet-mYLearnEstNNet).^2);
431 vTestNNetError = mean((mYTestNNet-mYTestEstNNet).^2);
432 fprintf('NNet mdl (Learning): %f\nNNet mdl (Test): %f\n', ...
433     mean(vLearnNNetError),mean(vTestNNetError));
434 
435 %Plot error per frequency increment
436 figure;
437 hold on;
438 plot(sDataDescription.vX,vLearnNNetError,'r');
439 plot(sDataDescription.vX,vTestNNetError,'g');
440 xlabel('Frequencies [Hz]');
441 ylabel('MSE');
442 title('NNet Model');
443 grid on;
444 legend('learning', 'test')
445 set(gca,'FontSize', 18);
446 
447 %% RUN STORED ARTIFICIAL NEURAL NETWORKS
448 mXLearnNNet = sDataFilt.mX(1:nLear,vVarMdl);
449 mYLearnNNet = sDataFilt.mX(1:nLear,vVarOut);
450 mXTestNNet = sDataFilt.mX(nLear+1:end,vVarMdl);
451 mYTestNNet = sDataFilt.mX(nLear+1:end,vVarOut);
452 [mXLearnNNet,muLearnNNet,sigmaLearnNNet] = zscore(mXLearnNNet);
453 mXTestNNet = (mXTestNNet-ones(size(mXTestNNet,1),1)*muLearnNNet) ...
454     ./(ones(size(mXTestNNet,1),1)*sigmaLearnNNet);
455 mANNet = nan(size(mXLearnNNet,2)+1,size(mYLearnNNet,2));
456 mYLearnEstNNet = nan(size(mYLearnNNet));
457 mYTestEstNNet = nan(size(mYTestNNet));
458 
459 %Load stored networks
```

```
460 netTmp = load('./NNmodel/Hidden20/cNet1');
461 netTmp = netTmp.cNet1;
462 clear netTmp.cNet1
463
464 %Run model
465 for idy=1:size(mYLearnNNet,2)
466     net=netTmp{idy};
467     mYLearnEstNNet(:,idy) = net(mXLearnNNet')';
468     mYTestEstNNet(:,idy) = net(mXTestNNet')';
469     %fprintf('%d/%d (%f)\n',idy,size(mYLearnNNet,2),toc(t00));
470     %keyboard;
471 end
472
473 %Model error
474 vLearnNNetError = mean((mYLearnNNet-mYLearnEstNNet).^2);
475 vTestNNetError = mean((mYTestNNet-mYTestEstNNet).^2);
476 fprintf('NNet mdl (Learning): %f\nNNet mdl (Test): %f\n', ...
477     mean(vLearnNNetError),mean(vTestNNetError));
478
479 %Plot model error
480 figure;
481 hold on;
482 plot(sDataDescription.vX,vLearnNNetError,'r');
483 plot(sDataDescription.vX,vTestNNetError,'g');
484 xlabel('Frequencies [Hz]');
485 ylabel('MSE');
486 title('NNet Model');
487 grid on;
488 legend('learning', 'test')
489 set(gca,'FontSize', 18);
490
491 %% REINTRODUCE TURBINE 37
492 if 1
493 %Input data for turbine 37
494 sDataFiltTmp = Filtering_AllTurb(sData);
495 vFlag = sDataFiltTmp.mX(:,1)==37;
```

```
496 mX37Tmp = sDataFiltTmp.mX(vFlag,:);
497 mX37 = zscore(mX37Tmp(:,vVarMdl));
498 mY37 = mX37Tmp(:,vVarOut);
499 mYLearnEstNNet37 = nan(size(mY37));
500
501 %Run Neural Network Model for turbine 37
502 for idy=1:size(mYLearnEstNNet37,2)
503     net=netTmp{idy};
504     mYLearnEstNNet37(:,idy) = net(mX37')';
505 end
506 %Calculate residuals
507   T37res = mYLearnEstNNet37 - mY37;
508 end
509
510 %% RESIDUAL MATRIX ANN MODEL
511 %Calculate residual matrix
512 mXTmp = [mYTestNNet-mYTestEstNNet ; mYLearnNNet-mYLearnEstNNet];
513 %Identify explained variance
514 expVar = sum(explained(1:2));
515 %Merge with auxiliary data
516 mXTmpAll =  [sDataFilt.mX(:,1:11) mXTmp];
517 [~, mu, sigma] = zscore(mXTmp);
518 [coeff,score,latent,tsquared,explained,mu]=pca(zscore(mXTmp));
519
520 %% RESPONSE FROM EACH TURBINE ONTO RESIDUALS
521 for idTurb=1:numel(vTurb)
522     figure;
523     vFlag = sDataFiltCentre.mX(:,1)==vTurb(idTurb);
524     plot(score(:,1),score(:,2),'.k');
525     %plot(score(:,1),score(:,2),score(:,3),'.k');
526     hold on;
527     plot(score(vFlag,1),score(vFlag,2),'.r');
528     title(sprintf('Turbine nr. %d',vTurb(idTurb)));
529     xlabel('Principal comp. 1')
530     ylabel('Principal comp. 2')
531     set(gca,'FontSize', 18);
```

```
532 end
533
534 %% RESPONS FOR EACH POWER BIN - ALL TURBINES
535 vBinUp = [1150 1403 1656 1909 2185 2415];
536 vBinLow =[921 1151 1404 1657 1910 2186];
537 vBin = unique(sDataFiltCentre.mX(:,3));
538 for idBin=1:numel(vBin)
539     vFlag = sDataFiltCentre.mX(:,3)==vBin(idBin);
540     subplot(2,3,idBin)
541     plot(score(:,1),score(:,2),'.k');
542     hold on;
543     plot(score(vFlag,1),score(vFlag,2),'.g');
544     title(sprintf('%d - %d kW %',vBinLow(idBin),vBinUp(idBin)));
545 end
546
547 %% WHAT HAPPENED TO TURBINE 37?
548 if 1
549 % Calculate PCA projection
550 mxTmp37norm = (T37res - ones(size(T37res,1),1)*mu)./ ...
551     (ones(size(T37res,1),1)*sigma);
552 mxTmp37norm = mxTmp37norm*coeff;
553 All =  [mX37Tmp(:,1:11) mxTmp37norm];
554 [~,vSort]=sort(All(:,11));
555 All = All(vSort,:);
556
557 %Plot response cluster
558 figure;
559 hold on
560 plot(score(:,1),score(:,2),'.k'); % plotter bakgrunnsrespons
561 plot(mxTmp37norm(:,1),mxTmp37norm(:,2),'.r'); % Plotter turbin 37
562 xlabel('Principal comp. 1')
563 ylabel('Principal comp. 2')
564 title('Turbine nr. 37');
565 set(gca,'FontSize', 18)
566 %Plot distance from center
567 figure;
```

```
568 vDateTmp = All(:,11);
569 vDistTmp = sum((mxTmp37norm(:,1:20)-ones(size(mxTmp37norm(:,1)))* ...
570     mean(mxTmp37norm(:,1:20))).^2,2);
571 plot(vDateTmp,vDistTmp,'.k-');
572 datetick('x','dd/mm/yy','keeplimits');
573 xlabel('Time');
574 ylabel('Distance from center');
575 title('Turbine nr. 37');
576 set(gca,'FontSize', 18)
577 end
578 %% CONDITION MONITOR
579 vUniTurb = unique(mXTmpAll(:,1));
580
581 for idt=1:numel(vUniTurb)
582     vInd = find(mXTmpAll(:,1)==vUniTurb(idt));
583     vDate = mXTmpAll(vInd,11);
584     [~,vOrd] = sort(vDate);
585     vInd = vInd(vOrd);
586     figure;
587     hold on
588
589 %     %3 dimensions
590 %     scatter3(score(:,1),score(:,2),score(:,3),'ko'); hold on;
591 %     scatter3(score(vInd,1),score(vInd,2),score(vInd,3),'r','filled');
592
593     % Response per turbine onto residual
594     plot(score(:,1),score(:,2),'.k');
595     plot(score(vInd,1),score(vInd,2),'.r');
596     title(sprintf('Turbine nr. %d',vUniTurb(idt)));
597     xlabel('Principal comp. 1')
598     ylabel('Principal comp. 2')
599     zlabel('Principal comp. 3')
600     legend('All turbines', sprintf('Turbine nr. %d' ,vUniTurb(idt)))
601     set(gca,'FontSize', 22);
602
603 %   %Calculate distance from center (> 2 dimensions)
```

```
604     vDistTmp = sum((score(vInd,1:20)-ones(numel(vInd),1)*...
605         mean(score(:,1:20))).^2,2);
606 %   %Calculate distance from center (< 2 dimensions)
607 %     vDistTmp = sqrt((score(vInd,1)-mean(score(:,1))).^2 ...
608 %         +(score(vInd,2)-mean(score(:,2))).^2);
609     vDateTmp = vDate(vOrd);
610     set(gca,'FontSize', 22)
611     %Plot distance from center
612     figure;
613     plot(vDateTmp,vDistTmp,'.k-');
614     hold on;
615     %Indicate cluster membership by color convention
616     plot(vDateTmp(score(vInd,1)>10),vDistTmp(score(vInd,1)>10),'or');
617     plot(vDateTmp(score(vInd,2)<-8),vDistTmp(score(vInd,2)<-8),'ob');
618     plot(vDateTmp(score(vInd,1)<0 & score(vInd,2)>8), ...
619         vDistTmp(score(vInd,1)<0 & score(vInd,2)>8),'og');
620     datetick('x','dd/mm/yy','keeplimits');
621     x=xlabel('Time');
622     y=ylabel('Distance from center');
623     t=title(sprintf('Turbine nr. %d',vUniTurb(idt)));
624     set(t,'FontSize', 20);
625     set(x,'FontSize', 20)
626     set(y,'FontSize', 20)
627     set(gca,'FontSize', 14)
628     pause(0.01)
629 end
```

Bibliography

Aldrich, C. and Auret, L. (2013). *Unsupervised Process Monitoring and Fault Diagnosis with Machine Learning Methods*. Advances in Computer Vision and Pattern Recognition. Springer London.

Antoniadou, I., Manson, G., Staszewski, W., Barszcz, T., and Worden, K. (2015). A time–frequency analysis approach for condition monitoring of a wind turbine gearbox under varying load conditions. *Mechanical Systems and Signal Processing*, 64–65:188 – 216.

Arlot, S. and Celisse, A. (2010). A survey of cross-validation procedures for model selection. *Statist. Surv.*, 4:40–79.

Ata, R. (2015). Artificial neural networks applications in wind energy systems: a review. *Renewable and Sustainable Energy Reviews*, 49:534 – 562.

Baxter, J. (2000). A model of inductive bias learning. *Journal Of Artificial Intelligence Research*, 12:149 – 198.

Blanke, M., Kinnaert, M., Lunze, J., and Staroswiecki, M. (2006). *Diagnosis and Fault-Tolerant Control*. Springer Berlin Heidelberg, Berlin, Heidelberg.

Chaari, F., Bartelmus, W., Zimroz, R., Fakhfakh, T., and Haddar, M. (2012). Gearbox vibration signal amplitude and frequency modulation. *Shock & Vibration*, 19(4):635 – 652.

Collacott, R. A. (1977). *Mechanical Fault Diagnosis and condition monitoring*. Springer Netherlands.

Cornelius, S. . P. G. (2004). *Machinery Vibration Analysis and Predictive Maintenance*. Elsevier, Linacre House, Jordal Hill, Oxford OX2 8DP 200 Wheeler Road, Burlington, MA 01803.

Dai (2014). *Safe and efficient operation and maintenance of offshore wind farms*. PhD thesis, Norwegian University of Science and Technology.

DESA (2015). World population prospects: The 2015 revision, key findings and advance tables. Technical report, United Nations, Department of Economic and Social Affairs.

Du, K.-L. and Swamy, M. N. S. (2014). *Neural Networks and Statistical Learning*. Springer London.

Faulstich, S., Hahn, B., and Tavner, P. J. (2011). Wind turbine downtime and its importance for offshore deployment. *Wind Energy*, 14(3):327–337.

GCube (2016). GCube Gets to Grips with Grinding Gearboxes. http://www.gcube-insurance.com/press/gcube-gets-to-grips-with-grinding-gearboxes/. [Online; accessed 02-June-2016].

GWEC (2014). Global wind report annual market update 2014. Technical report, Global Wind Energy Council, Rue d'Arlon 80 1040 Brussels, Belgium.

GWEC (2015). Global Wind Statistics issued by the Global Wind Energy Council. http://www.gwec.net/wp-content/uploads/vip/GWEC-PRstats-2015_LR_corrected.pdf. [Online; accessed 02-March-2016].

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning - Data Mining, Inference, and Prediction*. Springer Series in Statistics, second edition edition.

Hau, E. (2006). *Wind Turbines - Fundamentals, Technologies, Application, Economics*. Springer, Frühlingerstrasse 21 82152 Krailling Germany, 2nd edition.

Hau, E. (2013). Wind turbines - fundamentals, technologies, application, ecomics. Technical report.

He, G., Ding, K., Li, W., and Jiao, X. (2016). A novel order tracking method for wind turbine planetary gearbox vibration analysis based on discrete spectrum correction technique. *Renewable Energy*, 87, Part 1:364 – 375.

Ho, A., Mbistrova, A., and Corbetta, G. (2016). The european offshore wind industry - key trends and statistics 2015. Technical report, The European Wind Energy Association, Address: Rue d'Arlon 80, B-1040 Brussels, Belgium.

Hush, D., Abdallah, C., Heileman, G., and Docampo, D. (1997). Neural networks in fault detection: A case study. *Proceedings of the American Control Conference*, 2:918–921.

IEA (2013). Technology roadmap, wind energy. oecd/iea 2013. Technical report, International Energy Agency, 9 rue de la Fédération 75739 Paris Cedex 15, France.

IEC (1999). Wind Turbine Generator Systems. http://www.electropedia.org/iev/iev.nsf/display?openform&ievref=415-05-10. [Online; accessed 11-September-2015].

IEC61508. 1997. *Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems*. Technical report, International Electrotechnical Commission, Geneva.

Irena (2012). Wind power - renewable energy technologies: Cost analysis series. Technical Report 5/5, International Renewable Energy Agency, C67 Office Building, Khalidiyah (32 nd) Street P.O. Box 236, Abu Dhabi, C67 Office Building, Khalidiyah (32nd) Street, P.O. Box 263, Abu Dhabi, United Arab Emirates.

IRENA (2015). Renewable power generation costs in 2014. Technical report, International Renewable Energy Agency.

Isermann, R. (2011). *Fault-Diagnosis Applications: Model-Based Condition Monitoring: Actuators, Drives, Machinery, Plants, Sensors and Fault-tolerant Systems*. Springer Berlin Heidelberg.

IWES, F. (2014). Wind energy report germany 2014. Technical report, Fraunhofer Institute for Wind Energy and System Technology IWES, Königstor 59, 34119 Kassel.

Jardine, A. K., Lin, D., and Banjevic, D. (2006). A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing*, 20(7):1483 – 1510.

Kieffer, G. and Couture, T. D. (2015). Renewable energet target setting. Technical report, International Renewable Energy Agency (IRENA).

Kriesel, D. (2005). A Brief Introduction to Neural Networks. http://www.dkriesel.com/_media/science/neuronalenetze-en-zeta2-2col-dkrieselcom.pdf. [Online; accessed 20-May-2016].

Kumar, Y., Ringenberg, J., Depuru, S. S., Devabhaktuni, V. K., Lee, J. W., Nikolaidis, E., Andersen, B., and Afjeh, A. (2015). Wind energy: Trends and enabling technologies. *Renewable and Sustainable Energy Reviews*, 53:209 – 224.

Kusiak, A. (2016). Renewables: Shara data on wind energy. *Nature*, 529:19–21.

Kusiak, A. and Verma, A. (2012). Analyzing bearing faults in wind turbines: A data-mining approach. *Renewable Energy*, 48:110 – 116.

Laouti, N., Sheibat-Othman, N., and Othman, S. (2011). Support vector machines for fault detection in wind turbines. *{IFAC} Proceedings Volumes*, 44(1):7067 – 7072. 18th {IFAC} World Congress.

Leek, J. (2013). Cross validation. https://www.youtube.com/watch?v=CmEqvD_ov2o.

Lefebvre, N. (2015). Tmr 4260 - fault detection, isolation and prognosis. Lecture notes.

Lefebvre, N. (2016). Private communication. Conversation spring 2016.

Liu, H., qi Tian, H., Chen, C., and fei Li, Y. (2013). An experimental investigation of two wavelet-mlp hybrid frameworks for wind speed prediction using {GA} and {PSO} optimization. *International Journal of Electrical Power & Energy Systems*, 52:161 – 173.

Lund, B. F. (2016). Increasing wind farm profit through integrated condition monitoring and control. http://www.sintef.no/globalassets/project/eera-deepwind2016/presentations/closing_beritfloorlund_kongsberg_web.pdf. [Online; accessed 16-February-2016].

Madsen, B. (2011). Condition monitoring of wind turbines by electric signature analysis. Master's thesis, Technical University of Denmark, Copenhagen, Denmark.

Marquez, A. (2007). *The Maintenance Management Framework - Models and Methods for Complex Systems Maintenance*. Springer Series in Reliability Engineering.

MathWorks (2016). Statistics and Machine Learning Toolbox™ User's Guide. http://se.mathworks.com/help/stats/hierarchical-clustering.html. [Online; accessed 11-April-2016].

MATLAB (2015). *Statistics and Machine Learning Toolbox*. MathWorks, The MathWorks, Inc. 3 Apple Hill Drive Natick, MA 01760-2098, r2015b edition.

Ng, A. (2016). Principal Component Analysis Algorithm. https://www.coursera.org/learn/machine-learning/lecture/ZYIPa/principal-component-analysis-algorithm. [Online; accessed 04-April-2016].

Nie, M. and Wang, L. (2013). Review of condition monitoring and fault diagnosis technologies for wind turbine gearbox. *Procedia {CIRP}*, 11:287 – 290. 2nd International Through-life Engineering Services Conference.

Odgaard, P. F., Stoustrup, J., and Kinnaert, M. (2013). Fault-tolerant control of wind turbines: A benchmark model. *IEEE Transactions on Control Systems Technology*, 21(4):1168–1182.

Oljedirektoratet (1998). Basisstudie vedlikeholdsstyring. metode for egenvurdering av vedlikeholdsstyring. Technical report, Oljedirektoratet, Stavanger.

Pereira, F., Mitchell, T., and Botvinick, M. (2009). Machine learning classifiers and fmri: A tutorial overview. *NeuroImage*, 45(1, Supplement 1):S199–S209.

Pintelon, L. and Parodi-Herz, A. (2007). Maintenance: An evolutionary perspective. In Kobbacy, K. A. H. and Murthy, D. N. P., editors, *Complex System Maintenance Handbook*, chapter Maintenance: An evolutionary perspective, pages 21–48. Springer-Verlag London Limited, London.

Porotsky, S. and Bluvband, Z. (2012). Remaining useful life estimation for systems with non-trendability behavior. *Prognostics and Health Management (PHM), 2012 IEEE Conference on*.

Pozo, F. and Vidal, Y. (2016). Wind turbine fault detection through principal component analysis and statistical hypothesis testing. *Energies*, 9(1):3.

Ramírez, P. A. P. (2013). *Ageing management and life extension of technical systems*. PhD thesis, Norwegian University of Science and Technology.

Rausand, M. and Høyland, A. (2004). *System Reliability Theory: Models, Statistical Methods, and Applications*. Wiley, Hoboken, NJ, 2nd edition.

Redmon, N. (2002). A gentle introduction to the FFT. http://www.earlevel.com/main/2002/08/31/a-gentle-introduction-to-the-fft/. [Online; accessed 10-February-2016].

Rolfseng, J. (2015). Machine learning principles in operation and maintenance of wind turbines. Technical report, NTNU, Otto Nielsens veg 10 7491 Trondheim.

Salomonsen, J. E. (2015). Private communication. Conversation autumn 2015.

Salomonsen, J. E. (2015-2016). Email correspondence. Principal Engineer at Maintech AS.

Salvatore, J. (2013). World energy perspective - cost of energy technologies. Technical report, World Energy Council. Project Partner: Bloomberg New Energy Finance.

Samanta, B. (2004). Gear fault detection using artificial neural networks and support vector machines with genetic algorithms. *Mechanical Systems and Signal Processing*, 18(3):625 – 644.

Shafiee, M., Finkelstein, M., and Bérenguer, C. (2015). An opportunistic condition-based maintenance policy for offshore wind turbine blades subjected to degradation and environmental shocks. *Reliability Engineering & System Safety*, 142:463 – 471.

SIEMENS (2009). Product Calogue: Siemens Wind Turbine SWT-2.3-82 VS. http://www.energy.siemens.com/hq/pool/hq/power-generation/wind-power/E50001-W310-A123-X-4A00_WS_SWT-2.3-82%20VS_US.pdf. [Online; accessed 20-March-2016].

Silipo, R. (2015). Seven Techniques for Data Dimensionality Reduction. http://www.kdnuggets.com/2015/05/7-methods-data-dimensionality-reduction.html. [Online; accessed 16-April-2016].

Sperstad, I. B. (2015). Email correspondence. Researcher at SINTEF - Energy Systems.

Stopford, M. (2009). *Maritime Economics*. Routledge, 711 Third Avenue, New York, NY 10017, 3rd edition.

Suarez, J. (2009). Intro to Neural Networks. https://www.youtube.com/watch?v=DG5-UyRBQD4. [Online; accessed 07-April-2016].

Tchakoua, P., Wamkeue, R., Ouhrouche, M., Slaoui-Hasnaoui, F., Tameghe, T. A., and Ekemb, G. (2014). Wind turbine condition monitoring: State-of-the-art review, new trends, and future challenges. *Energies*, 7(4):2595.

Togstad, J. (2016). Private communication. Conversation medio March 2016.

Vachtsevanos, G., Lewis, F., Roemer, M., Hess, A., and Wu, B. (2006). *Intelligent Fault Diagnosis and Prognosis for Engineering Systems*. Wiley.

Valland, A., Lyridis, D. V., Reig, E., Giebhardt, I. B. S. J., Sørensen, J. D., Cradden, L., Malmo, O., Ojanguren, T., Pedersen, V. G. B., and deLaleu, V. (2014). Wp framework/industry challenges report - o&m; work package 4 - deliverable number 4.1. Technical report, SINTEF.

van Bussel, G. and Bierbooms, W. (2003). The dowec offshore reference windfarm: analysis of transportation for operation and maintenance. *Wind Engineering*, 27(5):381–392.

Vapnik, V. N. (1999). *The Nature of Statistical Learning Theory*. Springer-Verlag New York Berlin Heidelberg, Berlin, Heidelberg.

Verma, A. P. (2012). *Performance monitoring of wind turbines: a data-mining approach*. PhD thesis, University of Iowa.

Walpole, R. E. (2012). *Probability & Statistics for Engineers and Scientists*. Pearson, Boston, Mass, 9th edition.

Widodo, A. and Yang, B.-S. (2007). Support vector machine in machine condition monitoring and fault diagnosis. *Mechanical Systems and Signal Processing*, 21(6):2560 – 2574.

Yang, S., Li, W., and Wang, C. (2008). The intelligent fault diagnosis of wind turbine gearbox based on artificial neural network. In *Condition Monitoring and Diagnosis, 2008. CMD 2008. International Conference on*, pages 1327–1330.

Yang, W., Court, R., and Jiang, J. (2013). Wind turbine condition monitoring by the approach of {SCADA} data analysis. *Renewable Energy*, 53:365 – 376.