Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Engineering
Department of Mechanical and Industrial Engineering

Lisa Chanel Lien

Combined models for optimization of routing and maintenance scheduling for offshore wind farms

Master's thesis in Department of Mechanical and Industrial Engineering
Supervisor: Jørn Vatn
Co-supervisor: Wanwan Zhang
June 2022

Photo cred.: AEOLUS (https://aeolus-energy.com/)

NTNU
Norwegian University of
Science and Technology

Lisa Chanel Lien

Combined models for optimization of routing and maintenance scheduling for offshore wind farms

Master’s thesis in Department of Mechanical and Industrial Engineering
Supervisor: Jørn Vatn
Co-supervisor: Wanwan Zhang
June 2022

Norwegian University of Science and Technology
Faculty of Engineering
Department of Mechanical and Industrial Engineering

NTNU
Norwegian University of
Science and Technology

RAMS
Reliability, Availability,
Maintainability, and Safety

Combined models for optimization of routing and maintenance scheduling for offshore wind farms

Lisa Chanel Lien

June 2022

MASTER THESIS

Department of Mechanical and Industrial Engineering
Norwegian University of Science and Technology

(To be printed in color)

Supervisor 1: Jørn Vatn
Supervisor 2: Wanwan Zhang

Preface

This report is conducted in RAMS (reliability, availability, maintenance, and safety) at the Faculty of Engineering (IV). It is a part of my five-year study program in Engineering and ICT with a specialization in Production Management at NTNU. The thesis was written in the course TPK4930 - Production Management, Master's Thesis, during the spring semester of 2022.

This project was carried out as a part of the FME NorthWind program, with Jørn Vatn from the Department of Mechanical and Industrial Engineering at NTNU as my supervisor.

This report is recommended for readers who are master's students or have a higher degree with background knowledge in health management and maintenance optimization. It is beneficial if the reader has taken the following courses: TPK4120 - Industrial safety and reliability, TPK4161 - Supply Chain Analytics, and TPK4450 - Data driven prognostics and predictive maintenance.

Trondheim, 2022-06-11

Lisa Chanel Lien

Lisa Chanel Lien

Acknowledgments

I would like to acknowledge those who have contributed to the work related to this specialization project. First, I would like to give a warm thanks to my supervisor Jørn Vatn. Throughout the whole period of working on this project, we had weekly meetings and discussions. His prior knowledge and excellent guidance have contributed to giving me a good understanding of the background of the relevant topics. Furthermore, I would like to thank my co-supervisor Wanwan Zhang, RAMS PhD candidate, for support and valuable insight within predictive maintenance. Moreover, I would like to give a warm thanks to Vibeke Hvidegaard Petersen, IØT PhD candidate, and Andreas Lien, IØT Master's candidate, for support and valuable insight within the field of optimization. Additionally, I would like to thank my grandfather, David Bruns, for taking the time to proofread my entire project. Finally, I would like to thank Maximilian Georg Schuberth and Thibaut Forest from Equinor (Equinor Renewables, 2022) for valuable discussions and for providing me with real location and weather data from Dogger Bank. These key contributors have motivated me and helped me in the process of composing this master's thesis of which I am very proud.

L.C.L.

Remark:

Given the opportunity here, the RAMS group would recognize Professor Emeritus Marvin Rausand for the work to prepare this template. Some modifications have been proposed by Professor Mary Ann Lundteigen and Professor Jørn Vatn. In the preparation of this revised version important material from Associate Professor Anita Romsdal has also been included.

Executive Summary

During the last decades, offshore wind farms have been developing significantly and several new farms are being planned. This growth is mainly caused by the fact that the electricity production offshore is higher than onshore because the average wind speed is higher at sea. However, more resources and infrastructures are required to install and maintain offshore wind turbines, resulting in higher operation, installation, and maintenance costs for offshore wind farms. Operations and maintenance contribute to a high percentage of the total cost. These costs can be reduced by making the maintenance activities more efficient by optimizing maintenance vessel routing and maintenance schedules. Furthermore, predictive maintenance is used to maximize asset utilization by reducing changeover times and planned and unplanned downtime, and by increasing equipment performance. As the complexity of systems increases, the gap between existing optimization models for routing and scheduling, and predictive maintenance models will increase. So far, there seems to be no standardized method or proven technique to completely bridge this gap. Therefore, the main objective of this master's thesis is to suggest methods for combining optimization models for routing and scheduling with PdM models for offshore wind farm maintenance.

To reach the main objective, it is important to have knowledge within prognostics modeling and optimization modeling. The specialization project (Lien, 2021) focused on gaining knowledge and practical experience within the field of prognostics and degradation modeling. To gain knowledge about optimization models for routing and scheduling at offshore wind farms, a literature review is conducted. Furthermore, a simple optimization model for routing and scheduling has been created to gain some experience within this field. Before creating the optimization model, logistics data which was provided by an external company has been treated in various ways to find daily weather windows that comply with given constraints and to find the shortest path between all the turbines. The objective of this model is simply to minimize the number of days offshore at an offshore wind farm under certain constraints. An optimal result cannot be obtained within a reasonable amount of time when the model considers all the turbines at the farm and an entire year. Therefore, the model is divided into three and is sub-optimized instead.

After knowledge and experience were gained within both fields, the main objective could be tackled. A catalog comprising of four different approaches for combining optimization models with predictive maintenance models is presented. Then, a case study is made from one of these approaches. The results from this report's optimization model are used as the time windows for performing maintenance, and the predictive maintenance model from the specialization

project is used to create schedules based on the condition and the expected remaining useful lifetime (RUL) of each turbine. This stochastic programming problem considers a rolling horizon perspective when scheduling maintenance. The main risk is that the daily capacity may be exceeded because many turbines are postponed until the end of the maintenance period. The problem is that the stochastic parameters such as the expected weather and degradation trajectories might change along the way. To deal with this issue, four different heuristics are proposed to reduce the risk of exceeding the daily capacity on any given day. They still allow for a relatively flexible condition-based schedule so that healthy turbines can be postponed and impaired turbines can be maintained earlier. To reduce the total cost, the heuristics consider the wind turbines to be a part of a bigger network instead of considering them individually. The only varying factors between the four heuristics are the length of the period within which some turbines are required to be maintained, and the number of turbines that must be maintained within this period. The results from running the models for the four heuristics show that the suggested approach is better than taking a conservative approach and scheduling all turbines at the first and best opportunity one gets.

Sammendrag

I løpet av de siste tiårene har havvindparker utviklet seg betydelig og flere nye parker er under planlegging. Denne veksten er hovedsakelig forårsaket av at elektrisitetsproduksjonen offshore er høyere enn på land fordi gjennomsnittlig vindhastighet er høyere på havet. Det kreves imidlertid mer ressurser og infrastruktur for å installere og vedlikeholde offshore vindturbiner, noe som resulterer i høyere drifts-, installasjons- og vedlikeholdskostnader for havvindparker. Drift og vedlikehold bidrar til en høy prosentandel av totalkostnaden. Disse kostnadene kan reduseres ved å gjøre vedlikeholdsaktivitetene mer effektive ved å optimalisere vedlikehold fartøysruting og vedlikeholdsplaner. Videre brukes prediktivt vedlikehold for å maksimere ressursutnyttelsen ved å redusere overgangstider, planlagt og ikke-planlagt nedetid og ved å øke utstyrsytelsen. Ettersom kompleksiteten til systemene øker, vil gapet mellom eksisterende optimaliseringsmodeller for ruting og planlegging, og prediktive vedlikeholdsmodeller øke. Så langt ser det ut til at det ikke finnes noen standardisert metode eller utprøvd teknikk for å bygge bro over dette gapet. Hovedmålet med denne masteroppgaven er derfor å foreslå metoder for å kombinere optimaliseringsmodeller for ruting og planlegging, med prediktive vedlikeholdsmodeller for vedlikehold av havvindparker.

For å nå hovedmålet er det viktig å ha kunnskap innen prognostisk modellering og optimaliseringsmodellering. Spesialiseringsprosjektet (Lien, 2021) fokuserte på å tilegne seg kunnskap og praktisk erfaring innen feltet prognose og degraderingsmodellering. For å få kunnskap om optimaliseringsmodeller for ruting og planlegging ved havvindparker, gjennomføres først en litteraturgjennomgang. Videre er det laget en enkel optimaliseringsmodell for ruting og planlegging for å få litt erfaring innenfor dette feltet. Før optimeringsmodellen ble opprettet, har logistikkdataen som ble levert av et eksternt selskap blitt behandlet på ulike måter for å finne daglige værvinduer som samsvarer med gitte begrensninger og for å finne den korteste veien mellom alle turbinene. Målet med denne optimeringsmodellen er å minimere antall dager offshore på en havvindpark under visse begrensninger. Et optimalt resultat kan ikke oppnås innen rimelig tid når modellen tar hensyn til alle turbinene i parken og et helt år. Derfor er modellen delt i tre og suboptimalisert i stedet.

Etter at kunnskap og erfaring var opparbeidet innen begge felt, kunne hovedmålet angripes. En katalog som består av fire forskjellige tilnærminger for å kombinere optimaliseringsmodeller med prediktive vedlikeholdsmodeller presenteres. Deretter lages et casestudie fra en av disse tilnærmingene. Resultatene fra denne rapportens optimaliseringsmodell brukes som tidsvinduer for å utføre vedlikehold, og den prediktive vedlikeholdsmodellen fra spesialiseringsprosjektet brukes til å lage tidsplaner basert på tilstanden og forventet

gjenværende brukstid (RUL) for hver turbin. Dette stokastiske programmeringsproblemet tar et rullende horisontperspektiv i betraktning når vedlikeholdet planlegges. Hovedrisikoen er at den daglige kapasiteten kan bli overskredet fordi mange turbiner blir utsatt til slutten av vedlikeholdsperioden. Problemet er at de stokastiske parameterne som forventet vær og degraderingsbaner kan endre seg underveis. For å håndtere dette problemet foreslås fire ulike heuristikker for å redusere risikoen for å overskride den daglige kapasiteten på en gitt dag. De tillater fortsatt en relativt fleksibel tilstandsbasert tidsplan slik at friske turbiner kan utsettes, og syke turbiner kan flyttes tidligere. Heuristikkene vurderer vindturbinene som en del av et større nettverk i stedet for å vurdere dem individuelt for å redusere totalkostnaden. De eneste varierende faktorene mellom de fire heuristikkene er lengden på perioden hvor noen turbiner må vedlikeholdes innenfor, og antall turbiner som må vedlikeholdes innenfor denne perioden. Resultatene fra å kjøre modellene for de fire heuristikkene viser at den foreslåtte tilnærmingen er bedre enn å ta en konservativ tilnærming som planlegger alle turbiner ved den første og beste muligheten man får.

Contents

Preface . . . i

Acknowledgment . . . ii

Executive Summary . . . iii

Sammendrag . . . v

1 Introduction **2**

1.1 Background . . . 2

1.2 Objectives . . . 5

1.3 Research questions . . . 5

1.4 Research approach . . . 6

1.5 Contributions . . . 6

1.6 Limitations . . . 7

1.7 Outline . . . 7

2 Industrial background **8**

2.1 Wind turbine configuration . . . 8

2.2 Offshore wind farm configuration and installation . . . 9

2.3 Common failure modes of wind turbines . . . 10

2.4 Downtime after failure . . . 11

3 Maintenance, and prognostics and health management **14**

3.1 Maintenance and reliability . . . 14

3.2 Prognostics and health management . . . 15

3.2.1 Remaining useful life . . . 16

3.3 Prognostics and diagnostics . . . 16

3.4 Classification of models to predict RUL . . . 19

3.4.1 Physics-based models . . . 19

3.4.2 Data-driven models . . . 19

3.4.3 Hybrid models 20

4 Optimization and modeling 21

4.1 Problem formulation - modeling 21
4.2 Programming problems 24
4.2.1 Linear programming 24
4.2.2 Stochastic programming 26

5 Literature review 28

5.1 Routing and scheduling 28
5.2 Discussion 33

6 Optimization model for routing and scheduling 34

6.1 Treating the data 34
6.1.1 Distances between wind turbines 35
6.1.2 Weather windows for scheduling 36
6.1.3 Stochastic weather windows 37
6.2 Shortest path 38
6.2.1 Excel Solver 38
6.2.2 Python's mlrose package 39
6.2.3 Visual inspection 39
6.3 Mathematical formulation 40
6.3.1 Objective function 41
6.3.2 Constraints 41
6.4 Results 43
6.5 Discussion 44
6.5.1 Comparison to models from literature 44
6.5.2 Limitations to the proposed model 46

7 Optimization and predictive maintenance 48

7.1 Predictive maintenance model 48
7.1.1 Degradation model 48
7.1.2 Wiener process with linear drift 49
7.1.3 Cost function 50
7.1.4 Results 50
7.2 Catalog: Combining optimization and PdM models 51

7.2.1 Approach 1 51
7.2.2 Approach 2 52
7.2.3 Approach 3 53
7.2.4 Approach 4 54

8 Case study 56

8.1 Approach 2 56
8.1.1 Generating the degradation paths 56
8.1.2 Optimization problem 57
8.1.3 Explanation of algorithm 59
8.1.4 Results 61
8.1.5 Discussion of model and alternative solutions 62
8.2 Heuristics for Approach 2 63
8.2.1 Performance example 64
8.2.2 Explanation of algorithm 65
8.2.3 Results and discussion 68

9 Conclusions 70

9.1 Discussion 70
9.2 Summary and Conclusions 74
9.3 Recommendations for further work 76

A Acronyms 77

B Python code 78

B.1 Python code for optimization model in Chapter 6 78
B.2 Python code for Approach 2 in Section 8.1 83
B.3 Python code for Heuristics in Section 8.2 89

C Predictive maintenance model from specialization project 96

D Extra material for case study in Chapter 8 98

D.1 Expected costs and parameter values 98
D.2 Step-by-step example 99
D.3 Infeasible solutions from Approach 2 101

Bibliography 103

List of Figures

2.1 Main components in the nacelle of a horizontal-axis wind turbine system (Tong, 2010) . . . 9
2.2 Percentages of typical failures in wind turbines (Hahn et al., 2007) . . . 11
2.3 Failure frequency and downtimes of components (Hahn et al., 2007) . . . 13

3.1 Comprehensive diagnostic–prognostic process proposed by ISO13381 and implications on decision outcomes as well as required inputs (Sikorska et al., 2011) 18

6.1 Heat map of distances between all wind turbines in Dogger Bank A . . . 35
6.2 Schedule of number of hours per state per day from the year of 2018 . . . 37
6.3 Shortest path obtained by different solution methods . . . 39
(a) Solution method: Excel solver. Path length: 297 km . . . 39
(b) Solution method: Python mlrose. Path length: 1189 km . . . 39
(c) Solution method: Visual inspection. Path length: 178 km . . . 39

8.1 Example of degradation paths for 32 turbines in the optimal maintenance period from sub-model 1, Table 6.3 . . . 57
8.2 Expected costs of maintenance limits with varying lead times . . . 58
8.3 Flowchart of Approach 2. . . . 59
8.4 Degradation paths for 10 turbines, T#, in first period from sub-model 1, Table 6.3 . 61
8.5 Example visualization of the first period, d, and the restrictions, n, (for *Heuristic 1*) 64
8.6 Flowchart of Heuristics. . . . 66

D.1 Expected costs of lead times with varying maintenance limits . . . 98
D.2 Degradation paths for 32 turbines, T#, in first period from sub-model 1, Table 6.3 . 101

List of Tables

2.1 Typical failures in wind turbines (Gao and Liu, 2021) 12

4.1 References to literature on LP, MIP, and SP 24
4.2 The general form of LP- and MIP problems (Vatn, 2020b) 25
4.3 The general form of second, and first, second stage of SP problems (Vatn, 2020b) . 27

6.1 States of vessels that can be used based on the significant wave height requirements 36
6.2 Total number of hours per year of high wind and different states for three scenarios 38
6.3 Results from running the optimization model and the three sub-optimization models 44

8.1 The condition of the turbines at the time they got maintenance 61
8.2 Requirements for the different heuristics 63
8.3 Comparison of cost results from four different heuristics and a naive approach . . 69

C.1 Optimal maintenance limit for short, medium, long and stochastic lead times . . . 97

D.1 The parameter values used as a starting point for the case study. 99
D.2 Most cost-beneficial lead times according to cost equation 7.5 99
D.3 Daily conditions for example D.2 100
D.4 Comparison of results from proposed model to two relatively naive approaches . . 102

Chapter 1

Introduction

First, this chapter presents the background of the problem at stake. It is followed by the problem formulation, the main objectives to be solved, and the research questions. This chapter goes on to present the approach used to solve the objectives, the contributions, and the limitations associated with the project. Finally, a structural outline of the report is given.

1.1 Background

Offshore wind farms have been developing significantly during the last few decades, and several new farms are being planned. This growth is partially caused by the fact that the average wind speed is higher at sea than onshore, resulting in relatively higher electricity production rates offshore (Karlõševa et al., 2016). However, more resources and infrastructures are required to install and maintain offshore wind turbines, resulting in higher operation, installation, and maintenance costs for offshore wind farms. Scheu et al. (Scheu et al., 2012) state that operations and maintenance (O&M) can constitute one third of the total cost of energy from offshore wind generation due to long downtimes caused by poor accessibility in case of a failure. They point to offshore maintenance challenges such as relatively high failure rates, difficulties around scheduling, and harsh weather conditions and corresponding uncertainties making it hard to obtain maintenance windows. Other challenges may be lack of resources, and high costs of staff transportation, spare parts, and service vessels. However, Irwan et al. (Irawan et al., 2017) suggest reducing costs by making the maintenance activities more efficient by optimizing maintenance vessel routing and maintenance schedules. As offshore wind farms become larger and are constructed farther from shore, increasing the travel time of the vessels both to and within the farms, the potential for cost reduction becomes increasingly important. When scheduling the maintenance activities of an offshore wind farm there are several factors that need considering, such as the availability of various resources (e.g. service vessels, crews, and spare parts), weather conditions, and the disruption to electricity generation (Irawan et al., 2017). Irwan et al. suggest that the availability of a vessel and the weather conditions such as wave height and wind speed are the main factors that affect the performance of maintenance activities. For safety reasons, maintenance must only be performed within the weather windows, that is periods where the required weather conditions are met. Irwan et al. state that good weather periods are limited in most locations where offshore wind farms are located or being planned, and therefore maintenance schedules must be optimized to utilize the resulting weather windows.

Predictive maintenance (PdM) is one of the core value drivers in maximizing asset utilization and has gained attention within the industry and in academic fields (Wee et al., 2015). To reduce changeover times and planned and unplanned downtime, and to increase equipment performance, several industries are replacing their corrective maintenance (CM) strategies with preventive maintenance (PM) and PdM strategies. Prognostics and health management (PHM) is a central topic within PdM. PHM mainly focuses on predicting the future behavior and estimating the remaining useful life (RUL) of the item being considered (Vachtsevanos and Vachtsevanos, 2006). Several researchers such as Peng et al. (Peng et al., 2010), Sikorska et al. (Sikorska et al., 2011), and Gao et al. (Gao et al., 2015) all have different ways of classifying prognostic techniques, and there is no universal agreement of the classifications. This report classifies prognosis models into physics-based models, data-driven models, and hybrid models. Prognostics and RUL estimations need to be performed to implement PdM strategies, which requires knowledge about the item, available operation data, resources, and appropriate models for prediction. To enable real-time decision support and data analytics, relevant data must be extracted through sensors and monitoring technologies (Wee et al., 2015).

As the complexity of systems increase, the gap between existing prognostic models, optimization, and practice will increase. Up to this point, there seems to be no standardized method or proven technique to completely bridge this gap, and therefore future research is required within the field of PHM for successful implementation. Hofmann et al. (Hofmann, 2011) point out that in general the industry needs decision support tools to optimize maintenance execution and plan reduction of the maintenance costs and improve the availability of the wind turbines.

We have collaborated with Equinor and the first issue that came up was route planning for predefined annual routines as contract conditions require that a certain amount of maintenance is performed. One of the challenges lies in distributing the maintenance over a year given that each of the wind farms are installed during the summer months, which have the best weather conditions. Furthermore, it is interesting to look at how the predictive maintenance of the future can be included in the route planning. The following problem formulation has been given by Equinor (Equinor Renewables, 2022).

Problem Formulation

The Dogger Bank project consists of three wind farms: Dogger Bank A, Dogger Bank B, and Dogger Bank C, and each contains approximately the same number of wind turbines. The plan is to use one service operation vessel (SOV) to perform maintenance on all three wind farms. The SOV should visit each turbine around once per year. The three wind farms will be installed during the summer months in successive years, making it impossible to adhere to a strict one-year calendar based maintenance policy. This policy would make it so that all the turbines would have to be maintained by one SOV during the summer months. The main challenge is to schedule the order of maintenance visits such that it is spread out over the entire year, while not over- or under-maintaining some turbines. An important factor that needs to be considered is the weather windows within which maintenance can be performed. There are fewer weather windows during the winter months. In addition, the daily weather forecasts should be considered to minimize the total production loss. An SOV is equipped with a smaller, deployable vessel, a daughter craft (DC), meaning that there are two actors that need to be coordinated.

The goal of the study is to produce an algorithm that solves this problem through daily decision support. In addition, the study should be benchmarked against a deterministic approach and prove the upside in terms of less lost production and robustness. The case study uses several datasets and parameters for optimal configuration of the algorithm. The datasets contain 30 years of hindcast weather data such as the wind speed and wave heights for the relevant location, the location and installation schedule for the turbines, and the port location. The hindcast weather data is used to simulate daily forecasts. The parameters that are used are vessel speed, number of technician teams, number of maintenance hours per year per turbine, length of port call, length of shut down period before and after visit, and weather windows for the SOV and the DC. Equinor has said that there are four maintenance teams in total and that every turbine must get 20 hours of maintenance per year. The outputs of the model should be the drop-off and pickup times for turbines to be maintained on a given next day. There are three boundary conditions for the solutions which cannot be violated: (1) The SOV must return to port after 2 weeks at the farm for at least 6 hours in port. (2) A technician team must have 11 hours of rest period between deployments. (3) The DC cannot be more than 40 minutes travel distance away from the SOV. (Equinor Renewables, 2022)

The problem concerning PdM will be how to prioritize maintenance for the turbines based on their real-time condition and estimated RUL.

1.2 Objectives

The main objective of this master's thesis is to suggest methods for combining optimization models for routing and scheduling with PdM models for offshore wind farm maintenance. For this project, logistics from a company needs to be considered. There are several sub-objectives that need to be solved to reach the main objective:

- Objective 1: Present the general framework of wind turbines, offshore wind farms, typical wind turbine failures and critical components.
- Objective 2: Present the key concepts of PdM, PHM for RUL-estimation, and a classification of prognosis models.
- Objective 3: Present the key concepts of optimization and modeling.
- Objective 4: Perform an in-depth literature review on optimization models for routing and scheduling for offshore wind farms.
- Objective 5: Create a simple optimization model for routing and scheduling at offshore wind farms under logistics constraints and real data given by Equinor (Equinor Renewables, 2022).
- Objective 6: Create a catalog of ideas on how to merge optimization models with PdM models.
- Objective 7: Implement some of the ideas from the catalog through case studies.

1.3 Research questions

The following questions, *QX*, are the research questions. *Q1*: How do individual wind turbines and offshore wind farms work? *Q2*: What are the most common types of wind turbine failures, and what are their most critical components? *Q3*: How are prognosis and health management and RUL defined? *Q4*: What are the main ideas and steps in optimization and modeling? *Q5*: What optimization models have been made for routing and scheduling at offshore wind farms? *Q7*: How can aspects from the theory and literature be used to create a new simple optimization model for routing and scheduling? *Q8*: How can optimization and PdM models be merged?

1.4 Research approach

The aim of this thesis is to research methods to combine optimization and PdM models for creating good routes and schedules for maintenance at offshore wind farms. To gain understanding of the theory relevant for this report, two theory chapters are included: Chapter 3 covers topics concerning PdM and Chapter 4 cover topics related to optimization for routing and scheduling. To get a deeper understanding of how to create an optimization model for routing and scheduling for offshore wind farms, relevant literature has been studied. The main research platforms that have been used are ORIA, Web of Science, Google Scholar, ProQuest, and Scopus. The search-query used to find the articles used in the literature review in Chapter 5 was: "*routing and scheduling*" and "*stochastic programming*" and "*mixed integer*" and "*offshore wind*" and "*weather*". The selection criteria of the articles are based on the professional background of the authors, relevance of title, and the abstract. Before one can propose approaches to combine optimization and PdM models, it is important to gain some experience within both fields. The specialization project (Lien, 2021) used in preparation for the master's thesis provided valuable experience in creating a PdM model. Chapter 6 is dedicated to gaining some experience in creating an optimization model for routing and scheduling. To develop and test theory, valuable discussions have been held with Equinor (Equinor Renewables, 2022), and they have helped to identify challenges. This has been especially important to get realistic issues for route planning. Finally, some approaches for combining the two models are suggested, followed by a case study that tests out one of them with four heuristics.

1.5 Contributions

The main contribution of this project thesis is suggestions on how to combine optimization models for routing and scheduling with prognosis models for offshore wind farms maintenance. The optimization model that was created for routing and scheduling in Chapter 6 has resulted in a discussion that gives valuable insight into many of the challenges and uncertainties that come with optimizing routing and scheduling for offshore wind farms. This insight could be useful for someone who has also not studied optimization. Furthermore, section 7.2 presents different suggested approaches for combining the two types of models. The case study in Chapter 8 illustrates how optimization and PdM models can work together. This chapter first presents an algorithm on how maintenance activities are scheduled and updated in time. Then, it presents four heuristics which force a minimum number of turbines to be maintained within a given time frame to reduce the risk of the daily capacity being exceeded some days.

1.6 Limitations

The main objective of this project is to suggest methods for combining optimization models for routing and scheduling with PdM models for offshore wind farm maintenance. However, there is not much literature on how to combine such models. When searching for "predictive maintenance model" and "optimization model" and "offshore wind farm" there is only 1 result which is a review article found in Google Scholar. Therefore, a better understanding of both fields had to be obtained. Optimization is outside my field of study so time was required to understand this field. Furthermore, the resulting optimization model is rather simple, and it would take much more time to create a more advanced one. Equinor wanted a good strategy for planning maintenance for three offshore wind farms, with many turbines on each wind farm. It was difficult to find literature that focuses on solution methods for such big problems, so there was only time to study how to combine optimization and PdM models for one wind farm in the case study. Despite the optimization model being simple, the problem became too big when dealing with all the wind turbines at one farm. Therefore, the problem is sub-optimized with a reduced number of wind turbines to reduce the run time. If more time were available, more approaches from the catalog could be tested.

1.7 Outline

The report is comprised of eight chapters. Chapter 1 presents the background of this topic, main objectives, research questions, approaches, main contributions, and limitations related to the work. Chapter 2 presents the general framework of wind turbines, offshore wind farms, typical wind turbine failures and critical components. Chapter 3 presents the key concepts of PdM, PHM for RUL-estimation, and a classification of prognosis models. Chapter 4 presents a framework for problem formulation of optimization models and introduces some programming problems. Chapter 5 presents an in-depth literature review on optimization models for routing and scheduling for offshore wind farms. Chapter 6 uses the different steps from the problem formulation framework to create a simple optimization model for routing and scheduling, which is thereafter presented. Chapter 7 first describes a PdM model and then presents a catalog of suggestions on how to combine optimization models with PdM models. Chapter 8 presents a case study of modeling an approach from the catalog, and four tested heuristics for the approach. Finally, conclusions and future work are summarized in Chapter 9.

Chapter 2

Industrial background

This chapter first introduces the configuration of wind turbines. It goes on to give an overview of offshore wind farms. Then an overview of the most typical failure modes of wind turbines is given followed by an introduction of a critical wind turbine component and causes of downtime after failure.

2.1 Wind turbine configuration

Tong (Tong, 2010) gives multiple different classifications of wind turbines according to the: turbine generator configuration, airflow path relative to the turbine rotor, turbine capacity, the generator-driving pattern, the power supply mode, and the location of turbine installation. This report only introduces the turbine generator configuration and the location of turbine installation.

Wind turbines catch wind and transform it into electrical energy. Most of the larger, modern wind turbines are horizontal-axis turbines with three blades. Tong (Tong, 2010) classifies the rotating axis of the wind turbine blades as either horizontal or vertical. Horizontal-axis wind turbines have a rotating axis of blades that is parallel to the wind stream, while vertical-axis wind turbines rotate with respect to their vertical axis that is perpendicular to the ground. Some advantages with horizontal-axis wind turbines are high power density, high turbine efficiency, and low cost per unit power output. A wind turbine comprises a nacelle placed on top of a wind tower, housing turbine components inside, see Figure 2.1. Three blades are mounted on the rotor hub, which is connected to the gearbox via the main shaft. The rotor of the wind generator is connected to the output shaft of the gearbox. Therefore, the slow rotating speed

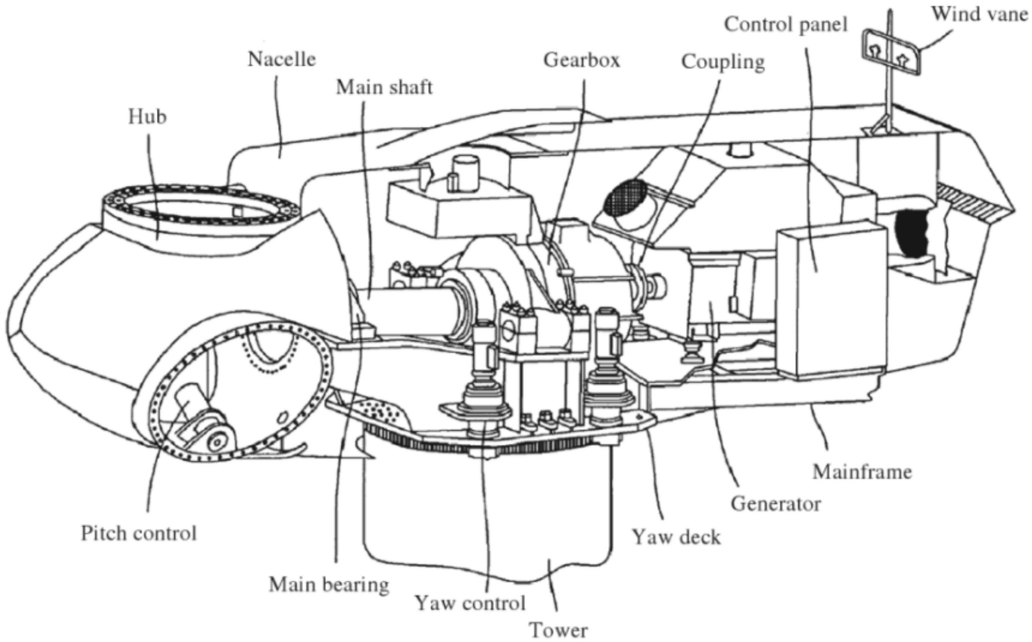


Figure 2.1: Main components in the nacelle of a horizontal-axis wind turbine system (Tong, 2010)

of the rotor hub is increased to a desired high rotating speed of the generator rotor. Each blade is pitched individually through the pitch control system to optimize the attach angle of the blade. This allows for a higher energy capture during normal operation and for protecting turbine components from being damaged during emergency situations. The yaw control system provides the yaw orientation control with the feedback information from the wind vanes, such as measured instantaneous wind direction and speed, for ensuring that the turbine is constantly facing against the wind. (Tong, 2010)

2.2 Offshore wind farm configuration and installation

The electricity generated from the turbines located in an offshore wind farm is fed to a single offshore substation and then transported to shore via high-voltage cables. Then, the electricity is transformed at an onshore substation before being fed into the public grid where it can reach the population (Ørsted, 2019). Most offshore wind farms are placed in locations where there is enough space to install numerous turbines, water depths are shallow enough to fix the turbines to the seabed, the turbines are above the crest level of the highest waves, and the wind conditions are optimal (Ørsted, 2019), (Wu et al., 2019). Chen and Blaabjerg (Chen and Blaabjerg, 2009) state that the ocean area is vast, which provides good conditions for developing large-scale wind farms. Wu et al. describe that most of the existing offshore wind turbines have a fixed foundation, but some of the turbines located in deeper waters must have a different foundation. This report does not further investigate the different foundations.

Wind energy is one of most promising clean energy sources and it is being widely used all over the world (Wang et al., 2021). Wang et al. show that the onshore wind power market installed capacity reached 54.2 GW and the offshore wind power market reached 6.0 GW in 2019. The onshore wind farms have a long history of development. Some of the advantages with onshore wind farms include lower cost in tower building and turbine installation, lower cost of foundations, easier integration with the electrical-grid network, and more convenient access for operation and maintenance (Tong, 2010). Nevertheless, the offshore wind industry has been developing faster than the onshore wind industry since the 1990s because of the outstanding wind power intensity and continuity at offshore sites (Tong, 2010). Kang et al. (Kang et al., 2019) explain that offshore wind farms receive more ocean wind than onshore wind farms, with longer availability per year, higher uniformity, and faster flowing speed. Karlõševa et al. (Karlõševa et al., 2016) discovered that the power generation is 50-100% higher at offshore sites than at onshore sites for the same types of wind turbines. Jeon et al. (Jeon et al., 2013) suggest that offshore wind turbines can protect the environment, and Hallowell et al. (Hallowell et al., 2018) suggest that they can save land resources. Tong states that environmental restrictions are laxer at offshore sites than at onshore sites due to factors such as turbine noise. Even though the offshore wind industry is developing, there are still several reliability and maintenance issues.

2.3 Common failure modes of wind turbines

Wind turbines are expensive to build and install, and certain failures can be dangerous and cause damage to people and to property. Performing maintenance at offshore wind sites can also be dangerous and complicated when scheduling around unpredictable weather windows. Maintenance at offshore wind farms comes with high costs partially due to the expensive vessels, which are also difficult to charter (Halvorsen-Weare et al., 2017).

Gao and Liu (Gao and Liu, 2021) state that wind turbine components are prone to failures or malfunctions because of either ephemeral events or aging degradation, which leads to economic losses and system interruptions. They categorize unexpected abnormal behaviors of wind turbines into failures and faults. van Schrick (van Schrick, 1997) defines faults as unpermitted deviations of at least one characteristic property of the system from the acceptable condition, and failures as the inability of a system to fulfill its function. Table 2.1 summarizes the main causes of typical failures. Figure 2.2 shows the reported downtime caused by

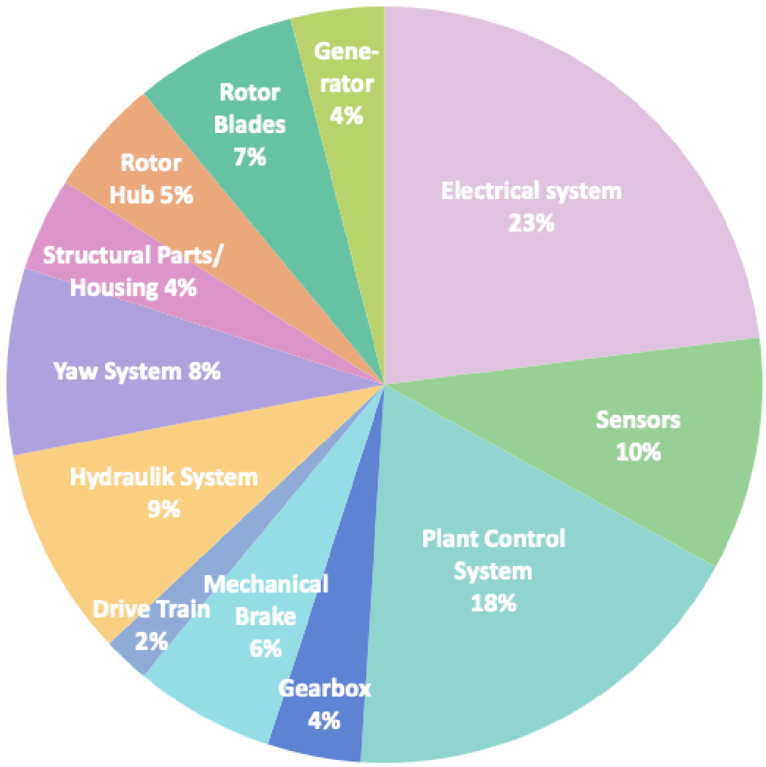


Figure 2.2: Percentages of typical failures in wind turbines (Hahn et al., 2007)

unforeseen malfunctions in wind turbines, which concerned half mechanical and half electrical components (Hahn et al., 2007). Defects, such as leakage or corrosion, can be discovered by discoloration of the component surfaces which can indicate small temperature variations or deteriorating condition (Márquez et al., 2012). Sound coming from turbine bearings can also indicate problems with their physical condition (Igarashi and Hamada, 1982). However, a more sophisticated approach to maintenance is required for several of the most typical failures such as electrical short circuits in the generator, cracks and roughness on the surfaces of the blades, and overheating of the gearbox (Márquez et al., 2012).

2.4 Downtime after failure

To describe the reliability of a machine, it is important to consider the downtime of the machines after a failure. The duration depends on necessary repair work, the availability of replacement parts, and on the personnel capacity of service teams (Hahn et al., 2007). Durstewitz and Wengler (Durstewitz and Wengler, 1998) report that in the past repairs to generator, drive train, hub, gearbox and blades have often caused standstill periods of several weeks. The average

Table 2.1: Typical failures in wind turbines (Gao and Liu, 2021)

Types of failures	**Causes of failures**
Failures on blades and rotors	Corrosion of blades and hub; crack; reduced stiffness; increased surface roughness; deformation of the blades; errors of pitch angle; and imbalance of rotors, etc.
Failures on gearbox	Imbalance and misalignment of shaft; damage of shaft, bearing and gear; broken shaft; high oil temperature; leaking oil; and poor lubrication, etc.
Failures on generator	Excessive vibrations of generator; overheating of generator and bearing; abnormal noises; and insulation damage, etc.
Failures on bearing	Overheating; and premature wear caused by unpredictable stress, etc.
Failures on main shaft	Misalignment; crack; corrosion; and coupling failure, etc.
Hydraulic failures	Sliding valve blockage; oil leakage, etc.
Failures on mechanical braking system	Hydraulic failures; and wind speed exceeding the limit, etc.
Failures on tower	Poor quality control during the manufacturing process; improper installation and loading; harsh environment, etc.
Failures on electrical systems/devices	Broken buried metal lines; corrosion or crack of traces; board delamination; component misalignment; electrical leaks; and cold-solder joints, etc.
Failures on sensors	Malfunction or physical failure of a sensor; malfunction of hardware or the communication link; and error of data processing or communication software, etc.

failure rate and the average downtime per component, as reported by Hahn et al., is given in Figure 2.3. Even though the figure is relatively old, it still illustrates that the high failure frequency of some components is to some extent balanced out by short standstill periods. Hahn et al. report that damages to generators, gearboxes, and drive trains have long downtime of about one week on average. Furthermore, researchers such as Sinha and Steel (Sinha and Steel, 2015), and Zhou et al. (Zhou et al., 2015) show that these components are extra critical due to long downtimes in case of failure. However, the work of Hahn et al. in Figure 2.3 does not differentiate between the contributions to downtime. The international standard NS-EN 13306 distinguishes between down time (DT), which is the time an item is down due to failure or other reasons, and time to restoration (TTR), which is time to recovery after failure. The mean time to restoration (MTTR) from NS-EN 13306 is used instead of MDT to clarify the time after a failure. Furthermore, MTTR can be divided into:

$$\mathrm{MTTR} = \mathrm{MLD} + \mathrm{MRT}, \tag{2.1}$$

where MLD is the mean logistic delay, which represents the waiting time for maintenance resources. MRT is the mean repair time, which represents the active time to complete the repair.

Then, the mean time between failures (MTBF) can be written as:

$$\mathrm{MTBF} = \mathrm{MTTR} + \mathrm{MTTF} = \mathrm{MLD} + \mathrm{MRT} + \mathrm{MTTF}, \tag{2.2}$$

which signifies the time between failures when we do not distinguish between uptime and downtime. This split is significant when it comes to combining optimization of routing and maintenance scheduling. The MRT is often governed by maintainability and is primarily the supplier's responsibility. Furthermore, the operator is in control of factors such as resource allocation and agreements for spare parts, which affect the MLD.

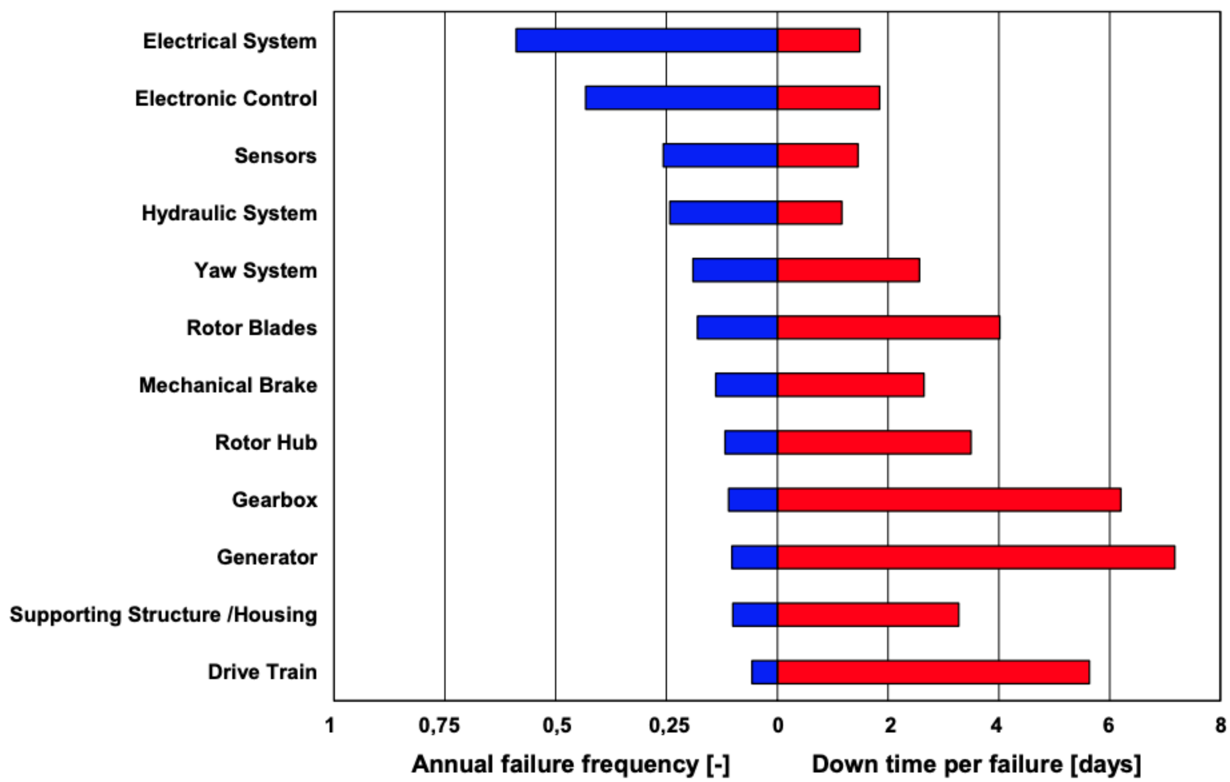


Figure 2.3: Failure frequency and downtimes of components (Hahn et al., 2007)

Chapter 3

Maintenance, and prognostics and health management

This chapter presents the concepts of PdM, PHM for RUL-estimation, prognostics and diagnostics, and a classification of prognosis models. The presentation is a summary of the specialization project (Lien, 2021), which is important to include because it creates a general basis for the PdM model which is used in Chapters 7 and 8.

3.1 Maintenance and reliability

Rausand et al. (Rausand et al., 2021) define maintenance as follows: "The combination of all technical and management actions during the life cycle of an item intended to retain the item in, or restore it to, a state in which it can perform as required". Maintenance is important to achieve high availability, which can be regarded as the reliability metric. Rausand et al. explain that availability measures the extent to which an item can operate at a future time t or during a future time interval (t_1, t_2). The availability of an item depends on certain factors such as recoverability (ability to recover from failure without repair), maintainability (level of difficulty for performing maintenance), and maintenance support (available resources for maintenance) (Rausand et al., 2021). The categorization of maintenance can vary. The European standards (BSI, 2010) (CEN, 2012) classify maintenance into CM and PM, while Rausand et al. categorize maintenance into CM, PM, and PdM. The latter categorization is presented in the following.

Corrective maintenance. The logic behind CM is to not fix or replace an item until it is broken. CM is known as run-to-failure or breakdown maintenance (Kim et al., 2017). The international standard (ISO13372, 2012) defines CM as "maintenance performed after a machine has failed". Rausand et al. (Rausand et al., 2021) state that CM "denotes all tasks that are carried out as a result of a detected item failure or fault, to restore the item to a specified condition". The goal of performing CM tasks is to bring an item back to functioning state in relatively short time by either replacing or repairing the failed part. Rausand et al. state that when a fault is revealed, CM tasks can either be carried out immediately or deferred, meaning that they are postponed until an opportunity occurs. Mobley et al. (Mobley, 2002) state that CM management averages about three times higher than a corresponding repair scheduled according to preventive mode.

Preventive maintenance. Rausand et al. (Rausand et al., 2021) describe that PM is maintenance that has been planned prior to the failure to reduce the degradation and the probability of failure. They explain that PM tasks usually involve inspection, parts replacement, adjustments, calibration, repair, and lubrication of items that are starting to wear out. Kim et al. (Kim et al., 2017) divide PM into time-based preventive maintenance (TBPM) and condition-based maintenance (CBM). Rausand et al. argue that PM tasks can be age-based, clock-based, condition-based, opportunity-based, overhaul, and degradation-based. Kim et al. explain that failures can be prevented through TBPM which sets periodic intervals regardless of the current health state of an item. TBPM is cost-effective if all components are expected to fail simultaneously. This is usually not the case, meaning that unnecessary replacements would take place. The international standard (ISO13372, 2012) defines CBM as "maintenance performed as governed by condition monitoring programmes", which means that maintenance is only performed when needed. Kim et al. express that this strategy can reduce the number of maintenance tasks, but it does not give any preparation time for the maintenance resources.

Predictive maintenance. Rausand et al. (Rausand et al., 2021) explain that PdM extends CBM by adding methods and theory which are used to predict the time when an item will fail. They express that PdM is based on prognosis for the degradation of the item. Prognostics is a key enabling technology for CBM to make a timely decision on maintenance and to schedule appropriate maintenance times (Kim et al., 2017).

3.2 Prognostics and health management

PHM is an engineering approach used to prevent equipment from failing unexpectedly by managing business risks to discover future outcomes in advance (Sikorska et al., 2011). PHM uses real-time health assessment of items under operating conditions and predicts the future state based on up-to-date information (Kim et al., 2017). PHM enables CBM by predicting the remaining life of an item during operation to reduce the total life cycle costs. The aim of

PHM is to actively anticipate the degradation of an item and schedule maintenance "just-in-time" (Kim et al., 2017). Kim et al. suggest that PHM is usually achieved by first constantly monitoring the health state of an item by using sensor measurements. Then, based on up-to-date measurements, data analytics algorithms can be used to predict the RUL of the item. Finally, a maintenance schedule should be developed accordingly to maintain the item in its original intended function. Kim et al. propose four main steps in PHM, which include: data acquisition, diagnostics, prognostics, and health management. Furthermore, the RUL is frequently used as a prediction indicator within PHM, indicating the RUL of the item at time t, based upon all the available information up to time t (Barros, 2019).

3.2.1 Remaining useful life

The estimation of the RUL is crucial within PHM. The RUL is usually an unknown and random variable and must therefore be estimated based on available health and condition monitoring data (Si et al., 2011). There are multiple definitions of the RUL such as the following two proposed by Si et al.; "The remaining useful life (RUL) of an asset or system is defined as the length from the current time to the end of the useful life" and "Remaining useful life (RUL) is the useful life left on an asset at a particular time of operation". By denoting $\mathrm{RUL}(t_j)$ as a random variable that corresponds to the RUL at time t_j, a general definition can be formulated:

$$\mathrm{RUL}(t_j) = inf\{h : Y(t_j + h) \in S_L | Y(t_j) < L, Y(s)_{0 \le s \le t_j}\}, \tag{3.1}$$

where $Y(t_j)$ denotes the condition of item j at time t_j and is related to diagnosis. $Y(t_j + h)$ denotes the future health state after time h and is related to prognosis. Furthermore, S_L denotes the set of unsatisfactory states of the item, while L denotes the fixed threshold limit which, if exceeded, defines the item as failed. A distribution of $\mathrm{RUL}(t_j)$ can be achieved in certain cases, meaning that confidence and uncertainty intervals related to the estimation can be achieved (Barros, 2019). This is often required considering that RUL is a stochastic variable. Having an understand of RUL is important in section 7.1.3 where a RUL-based cost function is introduced.

3.3 Prognostics and diagnostics

Sikorska et al. (Sikorska et al., 2011) state that "diagnostics involves identifying and quantifying the damage that has occurred", while "prognostics is concerned with trying to predict the damage that is yet to occur". These definitions suggest that prognostics relies upon the diagnostic outputs such as degradation rates and fault indicators. This is demonstrated in the general formulation denoting the RUL(tj) in equation 3.1 (Sikorska et al., 2011). This is used later, in section 8.1.1 where degradation paths are simulated to test a prognostics model. To obtain RUL-estimates and associated confidence limits, various steps from diagnostics to prognostics are required. Sikorska et al. exemplify this process by examining the process an item undergoes between a healthy state and final failure, by answering six questions. Questions (a) to (c) concern diagnostics, while questions (d) to (f) are in the realm of prognostics:

(a) Is a component in the degraded state?
(b) Which failure mode has initiated the degradation?
(c) How severe is the degradation?
(d) How quickly is degradation expected to progress from its current state to functional failure?
(e) What novel events will change this expected degradation behavior?
(f) How may other factors, such as the type of model and measurement noise, affect our estimate of RUL? (Sikorska et al., 2011)

The French standard (ISO13381-1, 2004) suggests a broader approach in describing the different steps in prognostics (Sikorska et al., 2011). The first step is *data pre-processing*, which includes diagnostics, failure definitions, identifying potential future failure modes, and selecting a suitable prognostic model. The second step is called *existing failure mode prognosis*, and it involves estimation of time to failure (ETTF) of all incipient failures and calculating the RUL of the item with the failure mode that has the lowest ETTF. This process is iteratively repeated until the RUL with the desired confidence limit is reached. The next step is described as *future failure mode prognosis*. This step involves assessing the most likely future failure modes and assessing the RUL with an appropriate confidence for all potential future failure modes. The final step is *post-action prognosis*, which involves identifying potential actions that can weaken, halt, or eliminate the progression of critical failure modes and prevent future failure modes. Then the previous modeling processes must be repeated with this information. A system-oriented approach to prognostics is needed to assess the impact the predicted failures have on maintenance and operational activities. A systems-oriented approach considers non-engineering factors such as inventory/supply management issues, maintenance planning options, and logistic concerns (Sikorska et al., 2011).

Sikorska et al. (Sikorska et al., 2011) developed a comprehensive flowchart that covers the steps involved in predicting the RUL. The flowchart is based on the steps suggested by the French standard (ISO13381-1, 2004), and illustrated in Figure 3.1. The first steps are related to fault detection and diagnostics and are expected to activate alarms notifying the user of system faults.

The next steps are related to prognostics and are divided into three levels that refer to steps from the ISO standard: **Level 1** refers to *existing failure mode prognosis*, **Level 2** refers to *future failure mode prognosis*, while **Level 3** refers to *post-action prognosis*. Each level builds on accurate and reliable outputs from the previous level. This implies that the modeling complexity increases with each level involved in realistic prognosis.

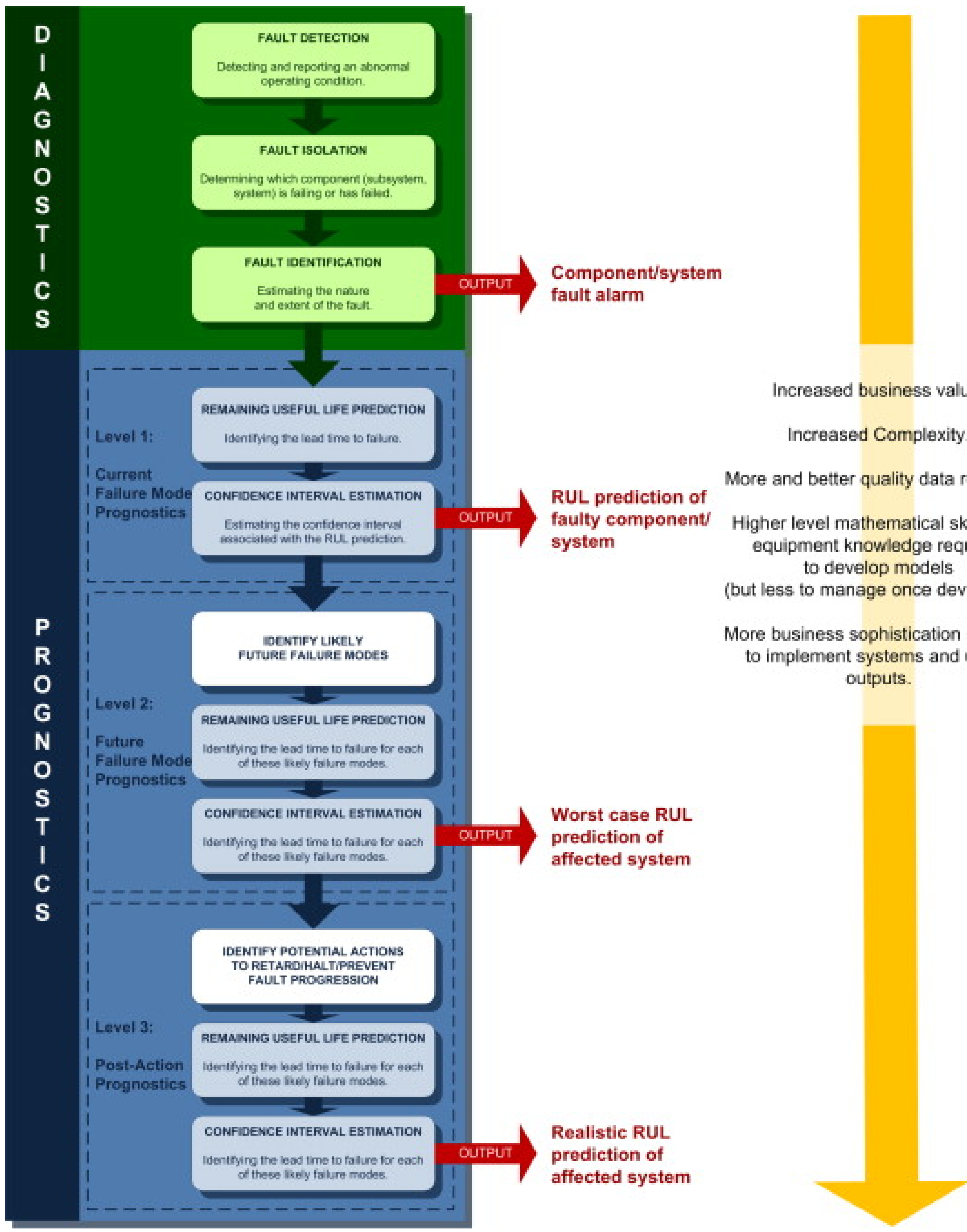


Figure 3.1: Comprehensive diagnostic–prognostic process proposed by ISO13381 and implications on decision outcomes as well as required inputs (Sikorska et al., 2011)

3.4 Classification of models to predict RUL

Researchers such as Peng et al. (Peng et al., 2010), Sikorska et al. (Sikorska et al., 2011), and Gao et al. (Gao et al., 2015) have all made an attempt to classify existing prognosis techniques used for estimating the RUL. Their classifications generally distinguish between physical, data-driven, and hybrid models, but no universal agreement has been reached on which model is the most optimal to use. This section introduces these three categories of prognostics approaches.

3.4.1 Physics-based models

Physics-based models use physical and mathematical expressions to describe degradation trends. They can be used to estimate the RUL of wind turbines by identifying the performance degradation from real-time condition monitoring. To establish a physical model of an item, detailed knowledge about the item behavior is required. This can however be difficult to obtain for multiple manufacturing systems. Gao et al. (Gao et al., 2015) state that physics-based models are often application-specific because their input parameters are obtained experimentally. They suggest using physics-based models for fatigue propagation and tool-life prediction. Tool-life prediction ties in machining parameters such as cutting speed, temperature, and feed rate, and these can be aggregated to a higher level for assessing machine performance. Physics-based models can be used for various degradation phenomena. However, they are often limited because linking complex systems to traditional physical models and obtaining knowledge about the physical degradation phenomena can be difficult.

3.4.2 Data-driven models

Gao and Liu (Gao and Liu, 2021) state that data-driven approaches use historical data and machine learning techniques to predict RUL for wind turbines, to train and learn system performance dynamics, and to identify current performance degradation from real-time data. Data-driven approaches can be divided into artificial intelligence approaches and statistical data-driven approaches. Data-driven models can be used in maintenance modeling to identify characteristics of the current health state of an item, and to predict the remaining time until the item fails. The two types of approaches are presented in the following.

Artificial intelligence approaches, such as machine learning, do not fit a physical or probabilistic model between the driving state and the failed state. This becomes helpful when the complexity of a system increases or when available data does not fit into probability models by specific distributions. Obtaining the uncertainty associated with machine learning estimations can be challenging because these approaches do not provide a distribution of the RUL. Si et al. (Si et al., 2011) suggest that “Business decisions based on prognostic information should [...] be based on the bounds of the RUL confidence interval rather than a specific value of expected life”. They argue that it is important to obtain uncertainty associated with RUL estimations in decision making to assess the risk associated with acting on the prediction. However, machine learning approaches estimate a modeling error from a validation data set to assess the uncertainty. This data is divided into a training data set used to perform the predictions, and a validation data set used to test the accuracy of the predictions. This will to some extent provide uncertainty related to the prediction (Barros, 2019), (Gao et al., 2015).

Statistical data-driven approaches assume a probabilistic distribution for the performance degradation or the fault deterioration. This approach is advantageous in that it results in a probability distributed prediction, implying that a confidence interval is given (Gao et al., 2015). Collected data from this approach can either be classified as event data (past recorded failure data) or condition monitoring data. Event data can be difficult to obtain for critical assets. Si et al. (Si et al., 2011) define condition monitoring data as “any data which may have a connection with the estimation of the RUL such as condition monitoring information, operational information, performance information, environmental information, and degradation signals”. It can be categorized as either indirect or direct. Si et al. state that “indirect condition monitoring data is the data which can only indirectly or partially indicate the underlying state of the system”, implying that event data might be required to estimate the RUL. Si et al. state that “direct condition monitoring data is the data which can describe the underlying state of the system directly so that the prediction of the RUL is actually the prediction of the condition monitoring data to reach a predefined threshold level”. This implies that additional event data will not be required, and the RUL can be estimated directly. For direct condition monitoring data, the Wiener processes is often considered (Si et al., 2011)(Gao et al., 2015), which is the model that is used in the PdM model in the case study of Chapter 8.

3.4.3 Hybrid models

Gao and Liu (Gao and Liu, 2021) write that hybrid approaches combine physics-based models and data-driven models for the formulation of degradation models and data analysis. This type of combination could reduce the complexity of calculation, help extract degradation data, or improve precision in predicting RUL.

Chapter 4

Optimization and modeling

Fundamental to solving optimization problems is to really understand the problem at hand, therefore this chapter lists important steps regarding problem formulation. Then it goes on to introduce programming problems, which are different tools that are used to structure the problem. Finally, this chapter goes further in depth of linear programming and stochastic programming, which are two types of programming problems.

4.1 Problem formulation - modeling

Vatn (Vatn, 2020b) explains that a description of a system using mathematical concepts and language is called a mathematical model. Furthermore, mathematical modeling is the process of developing a mathematical model. Mathematical models are useful for explaining systems, for studying the effects of different decisions, and for making predictions about future system behavior. Vatn states that it is common to distinguish between deterministic and probabilistic models. Deterministic models are mainly used to describe relations between physical quantities and other real-world observables. Probabilistic models, however, are not models of the real world, but are rather used to express uncertainty regarding observables in the real world. Probabilistic models enable analysts to perform probability calculus by applying the law of total probability in an efficient way when expressing uncertainty. (Vatn, 2020b)

Before creating an optimization model, it is important to work on the problem formulation. The different aspects that need to be considered are introduced in this following section, which is based on the work of Vatn (Vatn, 2020b).

Structuring. The first step required before the modeling starts is structuring. This step focuses on presenting aspects such as tactic knowledge and system understanding in such a way that the analyst can start modeling. Before creating the optimization model for routing and scheduling in Chapter 6, these aspects were considered and sketched out by pen and paper.

Identification of variables. The quantities in the model that vary are known as variables. There are two types of variables: decision variables and uncontrollable variables. The variables that the decision maker can control by implementing measures are known as decision variables. A decision variable could for example be to decide the number of items to order. However, problems can include quantities that the decision maker cannot influence, which are known as uncontrollable variables. Uncontrollable variables are often stochastic variables and could, for example, be the weather. Section 6.3 presents the decision variables and the uncontrollable variables of the optimization model created in this report. The only uncontrollable variable used in this model is the stochastic weather variable.

Determination of causes and effects. Variables are usually dependent in some way, and these dependencies must be specified.

Identification of model parameters. The fixed quantities in the model that describe variables or other parts of the model are called model parameters. Stochastic variables for instance are often described by their mean values and standard deviations. Section 6.3 presents the parameters of the optimization model created in this report.

Specify the objective function. The objective function is used in optimization problems, and it represents the function that should be minimized or maximized. Furthermore, it is a function of the decision variables. In many cases, the objective function is to minimize the costs of something. Equation 6.2 presents the objective function of the optimization model created in this report, which is to minimize the number of days that the maintenance vessel of an offshore wind farm must be offshore. The objective function only considers one binary decision variable. The reason for this is to create a simple optimization model. The different models studied in Chapter 5 all have more complicated objective functions which contain multiple decision variables.

Specify constraints. Optimization problems often contain constraints that must be dealt with. There might, for example, be limited resources or limitations to the production frequency of a machine. Section 6.3.2 presents all the constraints considered in the optimization model created in this report. For offshore wind farms, the constraints can typically include availability of technicians, spare parts, and vessels, environmental constraints such as wind speed and wave height, and legal constraints such as working hours, limitations to routing, and the speed of the vessels.

Modeling. Modeling is used to pool all the previous steps together in a consistent mathematical model. Usually, both deterministic and probabilistic models are required. The mathematical formulation of the optimization model created in this report is presented in section 6.3 and implemented with the code presented in Appendix B.1.

Identification of the need of data. Models require input data, which could for example be demand rates, production rates, and cost figures. The main objective of this step is to be specific on the need for data. However, it depends on the format and level of detail in the modeling.

Data collection and assessment of model parameters. After the need for data has been identified, the next step is to collect data and estimate or assign model parameters based on raw data and the use of expert judgments. Section 6.1 presents how the data used in the optimization model created in this report was treated. The result was three scenarios of weather windows as a basis for a simple, stochastic variable, which was used as a parameter in the model.

Run the model. The objective of modeling is to get insight into a complex problem area which might be achieved by running the model with different values. To search for good decisions, the model should be run with different values of decision variables. Section 6.3 shows that running the optimization model with all the turbines in the wind farm over an entire year takes very long. However, when the model is split into three sub-models the run time is significantly shorter, but the resulting objective value is the same.

Optimization. It is required to optimize the objective function in cases where an objective function has been specified. There exist various approaches which are introduced in the next section. (Vatn, 2020b). The optimization model proposed in this report is created and solved by using Gurobipy, which is a fast and powerful mathematical optimization solver.

Deb (Deb, 2014) describes that one can distinguish between single-objective optimization and multi-objective optimization in mathematical optimization. Multi-objective optimization is used when there are multiple goals for the optimization. Seyr and Muskulus (Seyr and Muskulus, 2019) exemplify this concept in the context of offshore wind farm maintenance where the goal can be to minimize the costs while maximizing the availability of the wind farm. It can be beneficial to use multi-objective optimization in this case since maximizing availability leads to a higher number of CM actions that cannot be grouped together, which will influence the maintenance cost.

4.2 Programming problems

This section is based on the work of Vatn (Vatn, 2020b). Programming problems deal with allocating resources such as materials, labor, machines, transportation capacity, energy, and capital, to minimize cost or maximize profit. Resources are in general limited. Vatn presents four types of programming: linear programming (LP), dynamic programming (DP), non-linear programming (NLP), and stochastic programming (SP). These are tools used to structure the problem so that it can be formulated explicitly in the form presented in section 4.1. In the following, linear programming and stochastic programming are further introduced since they are relevant for the upcoming chapters. Table 4.1 provides relevant literature for more in-depth readings on the relevant topics.

Table 4.1: References to literature on LP, MIP, and SP

Topic	**Reference to literature**
Linear programming (LP)	(Dantzig, 2016), (Chvatal et al., 1983), (Bertsimas and Tsitsiklis, 1997)
Mixed integer programming (MIP)	(Pochet and Wolsey, 2006), (Schrijver, 1998), (Nemhauser and Wolsey, 1988)
Stochastic programming (SP)	(Birge and Louveaux, 2011), (Shapiro et al., 2021), (King and Wallace, 2012)

4.2.1 Linear programming

Linear programming (LP) deals with problems defined by the following conditions:

1. The decision variables are non-negative.
2. The objective function is given as a linear function of the decision variables. The linear assumption implies that only the first powers of the decision variables are included, and no cross terms are allowed.
3. A set of linear equations or linear inequalities can be used to express constraints in terms of limitation of resources. (Vatn, 2020b)

Vatn states that a LP problem generally comprises an objective function $Z()$ to maximize or minimize, and a set of constraints $c_j x_j,\ j = 1, 2, ..., n$. It should be noted that the objective function is always of the decision variables x_j, but this is usually not explicitly stated in the $Z()$ function. Finally, all decision variables and some parameters must be non-negative. Table 4.2 shows the general form of LP problems written in matrix form, where $\mathbf{x}$ is a *decision vector*

Table 4.2: The general form of LP- and MIP problems (Vatn, 2020b)

	Linear programming	Mixed integer programming
Objective function	Maximize: $Z = \mathbf{cx}$	Maximize: $Z = \mathbf{cx}$
Constraints	$\mathbf{Ax} = \mathbf{b}$ $\mathbf{x} \geq \mathbf{0}$ $\mathbf{b} \geq \mathbf{0}$	$\mathbf{Ax} = \mathbf{b}$ $\mathbf{x} \geq \mathbf{0}$ $\mathbf{b} \geq \mathbf{0}$ x_j is an integer for $j \in \mathbf{I}$

controlled by the decision maker **c** which is the *profit/cost vector*. Finally, **A** is the *coefficient matrix* and **b** is the *requirement vector*. A LP problem can be solved by the SIMPLEX method, which was originally developed by George Dantzig (Dantzig, 2016). LP problems can be further studied in the work of Vatn (Vatn, 2020b). Vatn states that if LP problem formulations put additional logical constraints on the decision variables, the problem is a mixed integer programming (MIP) problem. The optimization model for routing and scheduling proposed in section 6.3 is a mixed integer problem. Therefore, MIP is introduced in the following section.

Mixed integer programming

Vatn (Vatn, 2020b) states that in MIP problems, some of the decision variables are typically restricted to integer values or even binary variables. A possible approach to MIP is to ignore the integer constraint, first solve the problem as an ordinary LP problem and then take the nearest integer from the continuous solution for those variables. However, this approach is rather naive and not particularly good. There is no general approach that guarantees the finding of an optimal solution because all possible combinations of the decision variables must be investigated, which is generally not possible. One of the most widely used methods for solving mixed integer problems in commercial computer codes is the branch-and-bound algorithm, which was originally developed by Land and Doig (Land and Doig, 1960). The branch-and-bound algorithm is an efficient enumeration procedure for examining all possible integer-feasible solutions. This algorithm can be further studied in the work of Land and Doig. Table 4.2 shows the general form of MIP problems, where **I** is the set of all integer variables. (Vatn, 2020b) The model proposed in section 6.3 is a MIP because it has binary variables such as deciding if the maintenance vessel is offshore on a given day or not.

4.2.2 Stochastic programming

This presentation of SP is based on Vatn (Vatn, 2020b). In standard LP problems all quantities are known in advance, making it a deterministic optimization problem. This means that $\mathbf{c}, \mathbf{A}$, and $\mathbf{b}$ are all known at the time of the decision to be made. In reality, decisions often have to be made when some of these quantities are unknown. Typically, the requirement vector $\mathbf{b}$ could be uncertain and would then need to be treated as a *stochastic variable*. This vector often represents resource constraints that may only or partially be known at a later stage in the decision process. Stochastic programming (SP) is a technique to solve such challenges. The following presentation limits SP to only deal with LP situations.

The following motivating example is borrowed from Vatn (Vatn, 2020b). Consider a fish processing facility where the production of tomorrow must be planned before the catch is known, i.e., the amount and quality of fish available for production. Fish of high quality can be used for either production of fresh fillet or frozen fish, while low quality fish can only be used for production of frozen fish. The selling price of fresh fillet is assumed to be higher than that of frozen fish, but production of fresh fillet requires more workload. The number of workers x to allocate for the production must be decided today, where x is commonly referred to as the here and now decision. The production profile y can be determined at a later stage when the catch is known. Now the objective function is split into two objective functions: Z_1 and Z_2. The *first stage* optimization is the maximization of $Z_1 = \mathbf{cx}$ subject to $\mathbf{Ax} = \mathbf{b}$, i.e., finding the number of workers to allocate for tomorrow's production. Assume that tomorrow's situation can be described by a stochastic vector $\mathbf{U}$. Given that the value of the vector is known, $\mathbf{U} = \mathbf{u}$, there is a second stage optimization problem which is given in Table 4.3.

The decision of tomorrow is influenced by the decision of today through the constraints $\mathbf{B}(\mathbf{u})\mathbf{x}$. This type of problem is known as the classical two stage linear stochastic programming problem, where the decisions of the two days are *interconnected*. In the first stage, the unknown result of the second stage decision must be considered, but this depends on the unknown vector $\mathbf{U}$. The first stage optimization problem adds the expected value of the best solution tomorrow to the objective function of today, see Table 4.3. The expectation is taken with respect to the stochastic vector $\mathbf{U}$ and $Q(\mathbf{x}, \mathbf{u})$ is the solution to the second stage optimization problem from the table.

The *profit/cost vector* $\mathbf{q}(\mathbf{u})$ and the *requirement vector* $\mathbf{d}(\mathbf{u})$ both depend on the value u that the stochastic vector $\mathbf{U}$ will take in the second stage problem. The *coefficient matrix* in $\mathbf{B}(\mathbf{u})\mathbf{x} + \mathbf{C}(\mathbf{u})\mathbf{y}$ is built up by two terms: $\mathbf{B}(\mathbf{u})$ relates to the first stage decision vector $\mathbf{x}$ and $\mathbf{C}(\mathbf{u})$ relates to the second stage decision vector $\mathbf{y}$. Both matrices depend on the value the stochastic vector $\mathbf{U}$ will take. From the example, $\mathbf{-cx}$ represents the cost of allocating x workers for tomorrows

Table 4.3: The general form of second, and first, second stage of SP problems (Vatn, 2020b)

	Second stage optimization problem	First stage (++) optimization problem
Objective function	Maximize: $Z_2 = \mathbf{q}(\mathbf{u})\mathbf{y}$	Maximize: $Z_{1,2} = \mathbf{cx} + \mathrm{E}_U[Q(\mathbf{x}, \mathbf{U})]$
Constraints	$\mathbf{B}(\mathbf{u})\mathbf{x} + \mathbf{C}(\mathbf{u})\mathbf{y} = \mathbf{d}(\mathbf{u})$ $\mathbf{y} \geq 0$ $\mathbf{d}(\mathbf{u}) \geq 0$	$\mathbf{Ax} = \mathbf{b}$ $\mathbf{x} \geq \mathbf{0}$ $\mathbf{b} \geq \mathbf{0}$

production, while $\mathbf{q}(\mathbf{u})\mathbf{y}$ represents the profit of tomorrow's production. Finally, $\mathbf{B}(\mathbf{u})\mathbf{x}+\mathbf{C}(\mathbf{u})\mathbf{y} = \mathbf{d}(\mathbf{u})$ gives the constraints of the number of workers and catch to consider. (Vatn, 2020b)

Later on, in section 6.3, a simple optimization model for routing and scheduling is presented. The first stage decision is what days the maintenance vessel should be offshore. This decision is made beforehand when the weather is uncertain. Then, when the vessel and maintenance crew are offshore and the weather is known, the second stage decision is to allocate the number of hours for performing maintenance on the different turbines. Alternatively, the first stage decision could have been what different parts and crew to bring on a trip. On the trip, things may have changed, and then the crew must optimize the use of resources they brought. This would be the second stage decision.

Scenario building

In this presentation, the uncertainty in the two-stage model is defined through the stochastic vector $\mathbf{U}$. The stochastic vector $\mathbf{U}$ could in general have many elements, where each element could take many different values, or even be continuous variables. This is challenging from a modeling and from a computational point of view because it will lead to a very large number of scenarios. There are many approaches to overcome this challenge. This report only presents one of them but refers to Vatn (Vatn, 2020b) for further readings. One approach that is often used is the sample average approximation (SAA). This approach uses Monte Carlo techniques to generate a limited sample of the stochastic vector $\mathbf{U}$. N vectors are realized, say $\mathbf{u}_i$, with the same direction as $\mathbf{U}$. The optimization problem to solve is then: (Vatn, 2020b)

$$\text{Maximize}: Z_{1,2} = \mathbf{cx} + \frac{1}{N}\sum_{i=1}^{N}\mathbf{q}(\mathbf{u}_i)\mathbf{y}_i \tag{4.1}$$

Later on, in section 6.1.3, three scenarios are created from three different years of weather data to account for some of the uncertainty around the weather. Since the model considers multiple scenarios, it performs more conservatively to find the most optimal global solution.

Chapter 5

Literature review

The literature for maintenance has already been reviewed in the specialization project (Lien, 2021). This chapter presents literature from optimization of routing and scheduling because both areas must be reviewed in order to combine them. This chapter first reviews some articles that are relevant for creating optimization models for routing and scheduling. Then it goes on to discuss how these models follow the suggested steps from section 4.1.

5.1 Routing and scheduling

To avoid downtime and production losses, an offshore wind farm requires frequent maintenance. The necessary maintenance resources such as helicopters and vessels are expensive and require optimization of routing and scheduling to use them effectively and to reduce the O&M costs. The search-query used to find the articles used in the following section is: *"routing and scheduling" and "stochastic programming" and "mixed integer" and "offshore wind" and "weather"*.

Dai et al. (Dai et al., 2015) introduce the routing and scheduling problem of a maintenance fleet for offshore wind farms (RSPMFOWF). In terms of cost, the goal of the problem is to determine the optimal assignments of turbines and routes to the vessels. They present the mathematical formulations for the RSPMFOWF that considers the limitations and characteristics of the problem. This optimization problem is solved by the Xpress Optimizer software using the branch-and-bound algorithm which is introduced in section 4.2.1. The model solves a multi-objective optimization problem, which is defined by Deb (Deb, 2014) in section 4.1. The model aims to minimize the total traveling costs of the vessels between the turbines and the total penalty cost of delayed days for maintenance tasks on the turbines. Dai et al. incorporate multiple constraints in their model. Each service vessel must leave and return the harbor only once per day, and every turbine is only visited once for delivery and once for pickup and must happen on the same day. However, if the vessel is required to be present during the maintenance operation on a turbine, the vessel will travel directly from delivery to pickup node. The time between delivery and pickup of personnel at a wind turbine must be greater than the required time to perform the maintenance. Dai et al. added a soft constraint to their model to make the performance of the maintenance task happen within the preferred time. The model makes sure that the vessels are not overloaded with equipment or personnel. In addition, the model restricts the number of working hours of the service vessels per day, and the decision variable that decides if the vessel is offshore or not is binary. The model of Dai et al. outputs the route that each scheduled vessel must complete on a given day. The results obtained by Dai et al. can be used directly in maintenance planning. However, the model proposed does not consider the uncertainty related to traveling time due to changing weather conditions, sudden emerging maintenance demand, and the out-of-stock situation.

Stålhane et al. (Stålhane et al., 2015) study the RSPMFOWF proposed by Dai et al. (Dai et al., 2015), but they only consider one period. They propose two alternative models to solve the problem: an arc-flow formulation, and a path-flow formulation. These are considered in further detail below.

The arc-flow model. The objective of the arc-flow model is to find a route through the network of wind turbines for each vessel, such that the total cost of performing the maintenance tasks is minimized. This model is a multi-objective optimization problem which aims to minimize the costs of transportation, downtime due to corrective and preventive maintenance tasks, and penalties for not performing maintenance tasks in time. The arc-flow model is solved with commercial software using the branch-and-bound algorithm which is introduced in section 4.2.1. There are multiple constraints to the model of Stålhane et al., such as limiting every maintenance task to be done at most once and ensuring a continuous vessel route between the turbines in the wind farm. Similarly to the model of Dai et al., this model makes sure that the weight capacity of the vessel is not exceeded with equipment and spare parts on board. Further, the pickup and delivery on a turbine must be performed by the same vessel. The model of Stålhane et al. keeps track of the number of technicians on board a vessel when leaving a given turbine, and the time at which each turbine is visited. Finally, some constraints enforce the delivery to happen before the corresponding pickup, and that the decision variable that describes if a vessel travels between two turbines must be binary. The arc-flow model outputs the number of nodes (or turbines) in the branch-and-bound tree given a time limit and a number of tasks and vessels.

The path-flow model. The path-flow model is reformulated from the arc-flow model by using Dantzig–Wolfe decomposition and is solved heuristically by generating a subset of the possible routes and schedules. The Dantzig–Wolfe decomposition method can be further studied in the article of Dantzig and Wolfe (Dantzig and Wolfe, 1960). This model is a multi-objective optimization problem which aims to minimize both the sum of the sailing costs and the penalty costs from not performing maintenance. Stålhane et al. incorporate multiple constraints in their model, such that a penalty cost occurs if all maintenance tasks have not been performed, and each vessel must sail one route exactly. Furthermore, the decision variables that describe if a vessel travels between two turbines and if a maintenance task has been performed in time must be binary. The path-flow model outputs the number of routes and schedules between the turbines that it was able to generate given a time limit and a number of tasks and vessels. Additionally, the difference between the path-flow model and the optimal solution found through the arc-flow model is always calculated to be within 1%.

Gundegjerde et al. (Gundegjerde et al., 2015) focus on reducing the maintenance costs by studying the vessel fleet size and mix problem that arises for the maintenance operations at offshore wind farms. They propose a stochastic three-stage programming model that considers uncertainty in weather conditions, failures to the system, vessel spot rates, and electricity prices. The topic of stochastic programming is introduced in section 4.2.2. This model is a multi-objective optimization model that aims to minimize the fixed costs of vessels and charted vessel bases (stage 1), the expected cost of the chartered vessels (stage 2), and the expected costs of transportation, using vessels, penalty, and downtime of delayed maintenance tasks (stage 3). Basically, decisions on the vessel fleet size and mix are made at stages 1 and 2, while the vessel fleet is deployed to execute maintenance activities at stage 3. The constraints at stages 1 and 2 restrict the number of vessels that can use a given onshore base, limit the investment in vessels and vessel bases, and determine the number of vessels going in and out of the fleet at the end of each rental period for vessels that can stay offshore for several periods. There are too many constraints for stage 3 to mention all of them, and therefore only some will be mentioned. All maintenance tasks are either completed within their time window or are not completed at all. The model of Gundegjerde et al. determines the number of vessels that are necessary to be acquired or chartered and the movement of vessels that can stay offshore for several periods between wind farms and their onshore base. Their model includes safety constraints that determine when maintenance activities cannot be executed due to rough weather conditions. Although there are many more constraints that have not been addressed, the main uncertain parameters are spot rates for vessels, weather conditions such as wind speed and wave heights, electricity prices, and failures. The aim of the model is to investigates the robustness of the solutions given limited time and different numbers of wind turbines in the wind farm. The results in the paper of Gundegjerde et al. show that their model can be used to solve problems for realistic-sized instances.

Stålhane et al. (Stålhane et al., 2016) propose a two-stage stochastic programming model, and this topic is introduced in section 4.2.2. The problem of interest is to determine the optimal fleet size and mix of vessels to support maintenance activities at multiple offshore wind farms. This is a multi-objective optimization model that aims to minimize the total investment cost and the fixed costs of the vessels in the fleet, the investment costs of offshore bases and their fixed costs, all costs for chartering in vessels, and the variable costs, such as fuel costs, for both chartering in vessels and purchased vessels. Furthermore, the model aims to maximize the value of the vessel fleet at the end of the planning horizon, the revenue for selling vessels, the revenue from chartering out vessels, and the reduction in revenue loss due to production stops. The model of Stålhane et al. has constraints that ensure the correct number of vessels from one period to the next and that the total investment in vessels and bases is limited to the budget. The number of vessels assigned to a base must not exceed the capacity of the base. Further, the model ensures that the O&M activities that must be performed will get performed, and the time spent on these activities is limited by the number of vessels in the fleet. The article of Stålhane et al. is an important addition to the literature as previous stochastic models have considered planning for only one year. The model considers uncertainty in demand and weather conditions and aims to consider the entire life span of the wind farm. The model outputs the expected value of perfect information, the value of the stochastic solution, the computational time, and the optimality gap, given the number of turbines, the number of periods, or the number of wind farms. The main challenge that Stålhane et al. face is that in making a more detailed representation of the tactical planning, the model will quickly become impractical to solve, which appears to be a challenging prospect for future research. Furthermore, the model should remain solvable for realistically sized problem instances.

Gutierrez-Alcoba et al. (Gutierrez-Alcoba et al., 2017) present a discrete optimization model that chooses optimal vessels to support offshore wind farm maintenance operations. They use bi-level optimization to handle real-time requests. On the first (tactical) level, the fleet composition is decided for a certain time horizon. On the second (operational) level, the operation schedule of the fleet is optimized, given failures and real weather conditions. Their model is a multi-objective optimization model which aims to minimize the fixed costs of operating the chosen bases, the charter cost of the available fleet of vessels during the entire planning horizon, the total cost of operating the bundles, the downtime costs of preventive and CM activity types, and the penalty costs incurred for PM and CM activities that are not performed in time. The only constraint for the tactical decision describes the usual relation that a given base should be in use to station vessels there. The constraints for the operational decisions limit the operations to the availability of sufficient vessels at each base, limit the operations based on available personnel, and assure that the first maintenance activity to be started is the first to be finished. There are other additional constraints, but they were confusing and are therefore not mentioned here. A case study is conducted to test the performance of the model, analyzing the optimal fleet of vessels given by the solver. The performance tests of the model of Gutierrez-Alcoba et al. show the complexity of the presented model for large time horizons.

Irawan et al. (Irawan et al., 2017) propose an optimization model and an efficient solution method based on decomposition methods. They are the first to introduce a solution method that deals with routing and scheduling for multiple wind farms and O&M bases. The model finds the optimal schedule for maintaining the turbines, the number of technicians required for each vessel, and the optimal routes for the crew transfer vessels. Irawan et al. develop an algorithm based on the Dantzig–Wolfe decomposition method (Dantzig and Wolfe, 1960). This algorithm solves a mixed integer linear program for each subset of turbines allowing it to generate all feasible routes and maintenance schedules for the vessels for every period. The routes consider multiple constraints such as weather conditions and the availability of vessels. Finally, they propose an integer linear program model to find the optimal route configuration and the optimal maintenance schedules. This model is a multi-objective optimization model that aims to minimize the maintenance costs which include travel, technicians, and penalty costs. This model considers multiple constraints such as weather windows to find the maximum number of available working hours, vessel personnel capacity, vessel load capacity, vessel availability, spare parts availability, the number of technicians available in the O&M base for each skill type, and the vessel ability to transfer spare parts. The model outputs maintenance costs (travel, technician, and penalty costs), a maintenance schedule for every wind turbine, routing for every vessel, and a technician schedule.

Schrotenboer et al. (Schrotenboer et al., 2020) introduce the stochastic maintenance fleet transportation problem for offshore wind farms (SMFTPO). The aim of the problem is to find the cost-minimizing distribution of maintenance tasks between vessels from the viewpoint of a maintenance provider that is responsible for multiple wind farms and must adhere to different contractual obligations. Schrotenboer et al. provide a two-stage stochastic mixed integer programming model for some SMFTPO settings and solve it by using sample average approximation. Both stochastic programming and mixed integer programming are introduced in section 4.2. Their model controls for uncertainty in the weather conditions and maintenance tasks.

5.2 Discussion

From section 4.1, all the presented articles follow all the suggested steps. The first step concerning *structuring* was performed somewhat differently in the different articles. Dai et al. (Dai et al., 2015), Gundegjerde et al. (Gundegjerde et al., 2015), Irawan et al. (Irawan et al., 2017), and Schrotenboer et al. (Schrotenboer et al., 2020) all presented illustrations with descriptions of how the systems work. The other authors did it verbally. Most of the articles present the *identified variables* and *model parameters* in clear lists, except for Stålhane et al. (Stålhane et al., 2015) who present them in the text. Furthermore, all the articles presented some *determination of cause and effect*, some more detailed than others. The *objective function* and *constraints* are specified in all the articles, and then all articles perform the *modeling* step by pooling all the previous factors into clear mathematical models. Furthermore, all the articles *identified the need of data* and *collected data*. Most of the articles have used data from other research articles or online databases. Finally, all the articles have *run the model* and *optimized* the objective function of their models. The articles all presented some numerical results in the last chapter before their conclusion.

Chapter 6

Optimization model for routing and scheduling

In this chapter, an attempt has been made to follow the steps regarding problem formulation from section 4.1. This chapter begins with explaining how the data that is being used in the optimization model for routing and scheduling is treated. It goes on to demonstrate some challenges around finding the shortest path in a problem containing many visitation nodes. Furthermore, a mathematical formulation of the optimization model suggested in this report is presented, followed by the results of running the model. Then, the model is compared to the models presented in the literature review. Finally, a discussion of the model is carried out.

6.1 Treating the data

Before creating an optimization model, it is important to work on the problem formulation. This section focuses on the *data collection and assessment of model parameters* step from section 4.1. Equinor provided 30 years of weather data containing information about time, significant wave height, peak wave period, wave period T01, wave period T02, peak wave direction, mean wave direction, wind speed (10m), wind direction (10m), current speed, and current direction. The data is given hour by hour for one specific coordinate in between the three wind farms. In addition, they provided data containing the location of their assets, i.e., of the wind turbines and the office: Port of Tyne. The data is divided into wind farm, asset type, asset name, easting and northing (UTM zone 31N), longitude and latitude (based on WGS 84), and depth below mean sea level. All of the provided data is in Excel files. The data has been treated in multiple different ways which the following sections describe in further detail.

6.1.1 Distances between wind turbines

UTM coordinates are given as a zone number together with an easting-northing pair. Given two points $t_1 = (E_1, N_1)$ and $t_2 = (E_2, N_2)$ in the same zone, the distance between these two points is well approximated by the Euclidean distance between the easting and northing coordinates (Cohen, 2004):

$$d(t_1, t_2) = \sqrt{(E_1 - E_2)^2 + (N_1 - N_2)^2}, \tag{6.1}$$

A table of the distance between the turbines in Dogger Bank A was created. It contains 9216 (from 96 wind turbines) data points with the distance, in kilometers, from one turbine to another. Figure 6.1 shows a heat map of the distances between all wind turbines in the wind farm (green = short distance, yellow = medium distance, red = long distance). The distances in the table are too small to see due to there being too many numbers, however, each individual number is not important for this report. Furthermore, since all of the data points were given in Excel, it seemed like the most efficient solution to further treat them in Excel. The sequence of the turbines in Figure 6.1 is given by the sequence of the data provided by Equinor. As seen in the figure, there is a diagonal line of green from the upper-left corner to lower-right corner. These distances are equal to zero since they show the distance from each turbine to itself. The shortest distance from one turbine to the next in the wind farm is 1.4 km, while the longest distance is 37.0 km.

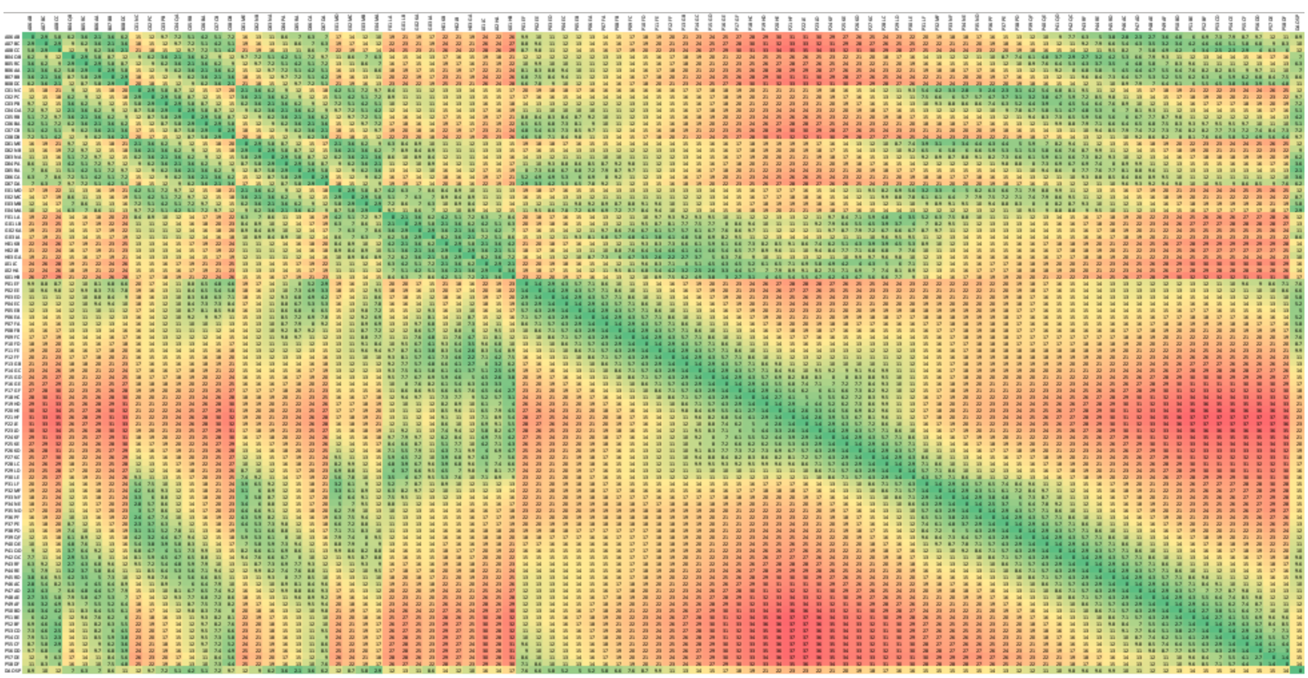


Figure 6.1: Heat map of distances between all wind turbines in Dogger Bank A

6.1.2 Weather windows for scheduling

To consider the weather windows for the scheduling, the wave height and the wind speed have been considered. The following sections go further in detail on how the data is treated.

Weather windows with respect to wave height

Equinor gave certain requirements for when the different vessels could be used and not. The SOV, or mother craft, can only be used if the significant wave height is below 3.0 meters. If the significant wave height is below 1.2 meters, the daughter craft (DC) can be used additionally. Table 6.1 shows what the different states are and what color is used to describe the states. These criteria have been used to find hourly states of the weather data from 2018. Figure 6.2 shows the number of hours per state per day in 2018, which is used to create the weather windows for a deterministic model. To simplify the deterministic model, only one state is assumed every day. The daily state has been decided by the following criteria:

1. State 2 will only be used if there are more than 10 hours of weather window of that state.
2. The longest coherent weather window of a state above zero will be used per day.
3. If state 1 is used, the coherent hours of state 2 weather window are also considered to be state 1.
4. There is no weather window if the longest coherent weather window of a state above zero is less than 3 hours.

Another assumption that was made to simplify the deterministic model was that a total of one hour of the daily weather window is wasted in traveling time in state 2, while in state 1, 2 hours are wasted in traveling time.

Table 6.1: States of vessels that can be used based on the significant wave height requirements

States	Description	Color
0	Significant wave heigh >3.0 meters. No vessels can be used.	
1	3.0 meters >Significant wave height >1.2 meters. Only the SOV can be used.	
2	1.2 >Significant wave height. Both the SOV and the DC can be used.	

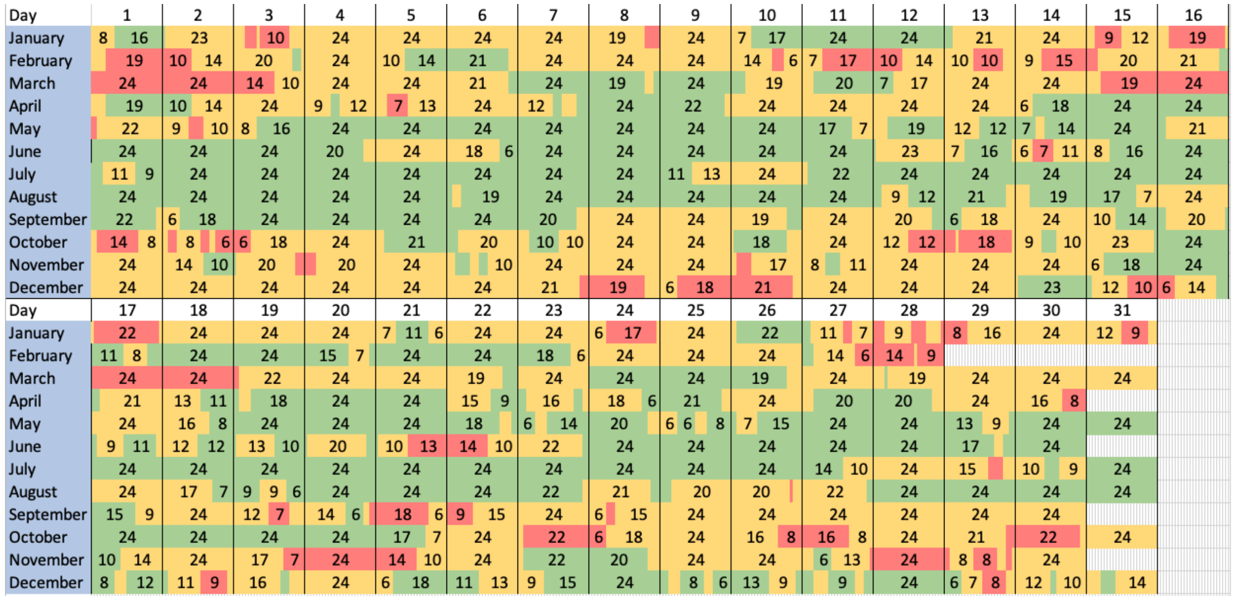


Figure 6.2: Schedule of number of hours per state per day from the year of 2018

Weather windows with respect to wind speed

The data of 2018 showed that in less than six percent of the time (hour by hour), the wind speed is above 14 meters per second = 50.4 kilometers per hour. Equinor (Equinor Renewables, 2022) wishes to not do maintenance if the wind speed is high, and therefore a wind speed of 14 m/s is assumed to be the fixed threshold for not doing maintenance. In Excel, the value 1 is assigned to the hours of high wind speed, while 0 is assigned to the hours of low wind speed, for all hours in 2018. Then the sum of these values is calculated for every day to gain knowledge of the number of hours with high wind speed per day. It is assumed that if there are more than four hours of high wind speed a day, maintenance will not be performed that day. The weaknesses of this assumption are that the hours of high wind speed might not be consecutive, and they might also be in the middle of the night.

6.1.3 Stochastic weather windows

To account for the unpredictable weather windows, three scenarios have been made from the weather data. Scenario building is introduced in section 4.2.2. Table 6.2 shows the years that have been used as scenarios, the number of hours they are in the different states, and the number of hours they have high wind speed. The data is formatted ahead of time with respect to significant wave height and wind speed. From the 30 years of wind data, 1990 was the year with the highest number of hours in state 0. This means that there were fewer and shorter weather windows that year. In addition, the number of hours of high wind speed is the highest in 1990, which supports the previous statement. The year of 2003 was picked as a scenario because it had the highest number of hours in state 2, meaning that there were more and longer weather windows that year. This statement is supported by the relatively low number of hours with high wind speed. The three years from Table 6.2 work well as the three scenarios in the stochastic optimization model because they are quite different and demonstrate that the length and the amount of the weather windows can vary from year to year. The number of daily available hours for these three scenarios can be found in Appendix B.1 in the following python lists: "weather_window_1990", "weather_window_2003", and "weather_window_2018". The first element in each list represents the first day of the year, the second element represents the second day of the year, and so on. Originally, there were 365 elements per list, but they have been shortened down in the appendix.

Table 6.2: Total number of hours per year of high wind and different states for three scenarios

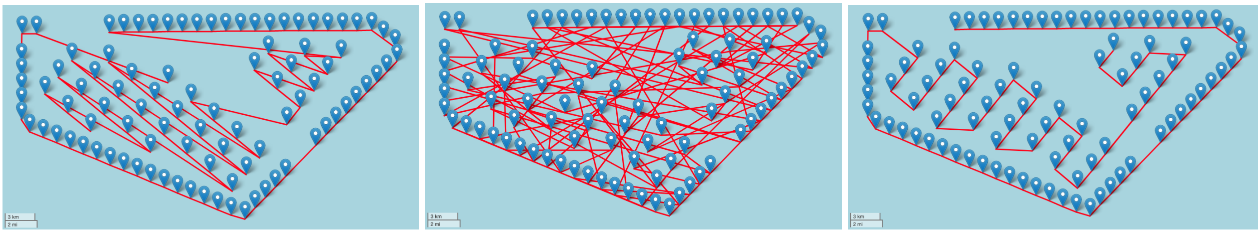
Year	State 0 Hours	State 1 Hours	State 2 Hours	High wind Hours
1990	1294	4287	3179	817
2003	597	4211	3952	318
2018	821	4181	3758	492

6.2 Shortest path

The problem under consideration is finding the optimal path between all wind turbines on Dogger Bank A such that the total travel distance is minimized.

6.2.1 Excel Solver

This problem should theoretically be solvable in the built-in solver tool in Excel. The first step is to create an initial sequence where every wind turbine is visited once. The problem is to find a sequence such that the total distance traveled is minimized. The distance from one turbine to the next in the initial sequence can be found using the built-in index function in Excel. After this step is done for all turbines, an initial proposal for the shortest distance is created. The second step is to use the sum function in Excel to find the total travel distance of the initial sequence. Finding the optimal sequence can be done by using Excel Solver. The objective is to minimize the total travel distance between the turbines by varying the sequence which consists of the variable cells. The only constraint for this problem is that every turbine is only visited once.



(a) Solution method: Excel solver. Path length: 297 km

(b) Solution method: Python mlrose. Path length: 1189 km

(c) Solution method: Visual inspection. Path length: 178 km

Figure 6.3: Shortest path obtained by different solution methods

Then the problem should theoretically be solvable using the evolutionary solver. However, this tool is not strong enough to find the optimal solution. Excel Solver was not able to find a better sequence than the initial sequence. Therefore, due to luck, this might be the reason for why Excel Solver was able to find a relatively good solution as illustrated in Figure 6.3 (a).

6.2.2 Python's mlrose package

This solution method is based on the work of Hayes (Hayes, 2019). A solution to finding the shortest path could be found by defining an optimization problem object that only considers valid tours of the turbines as potential solutions, which can be implemented in mlrose. The steps suggested by Hayes to solve this problem are: (1) Define a fitness function object, (2) Define an optimization problem object, and (3) Select and run a randomized optimization algorithm. After the script for the solution was written and run in Python, the solution came out looking like Figure 6.3 (b). It becomes obvious that this program was far away from finding the best solution.

6.2.3 Visual inspection

Figure 6.3 (c) shows the sequence found by using visual inspection which shows that finding a sequence by hand can in some cases be easier and give better results than a computer or an algorithm can. The descriptions in Figure 6.3 show the length of the shortest paths obtained by the different solution methods. The shortest distance found by Python's mlrose package is more than six times as long as the shortest distance found by visual inspection.

Finding the shortest path, or the Traveling Salesman Problem, is a NP-hard problem. In this case it means that it is not feasible to evaluate every possible problem solution within a reasonable time when the number of wind turbines are large (Hayes, 2019). In fact there are $96! = 9.916779 \cdot 10^{149}$ possible solutions to this problem. Even though the solution obtained using mlrose might seem bad, there exist even worse solutions.

6.3 Mathematical formulation

Following is a presentation of the stochastic optimization model for determining the optimal schedule for maintenance at the offshore wind farm, Dogger Bank A. Two of the problem formulation steps from section 4.1, are *identification of model parameters* and *identification of variables*. The results of these two steps are presented in the following, through symbols used to describe the mathematical optimization model.

Parameters

W	Set of all wind turbines at Dogger Bank A, $W = \{w_1, w_2, ..., w_{96}\}$
D	Set of all days in a year, $D = \{d_1, d_2, ..., d_{365}\}$
S	Set of scenarios of wind data, $S = \{s_1, s_2, s_3\}$
A_{ds}	Total number of available hours for maintenance on day d in scenario s, considering number of vessels, wave height, and wind speed
N	Number of maintenance teams
T	Number of maintenance hours required per turbine w
C	Maximum number of hours a maintenance team can work per day
M	Big m which is a method of solving linear programming problems by using the simplex algorithm

Decision variables

x_d	Binary variable that says if the maintenance vessel is offshore on day d or not, $x_d \in \{0 : \text{onshore}, 1 : \text{offshore}\}$
y_{dws}	Number of maintenance hours to be performed on turbine w on day d in scenario s
z_{dws}	Total number of maintenance hours that have been performed on turbine w up to day d in scenario s
q_{dw}	Ensures the sequence of turbine w on day d
p_d	Ensures that the maintenance vessel is 14 consecutive days offshore
h_d	Ensures that the maintenance vessel is 28 consecutive days onshore

This model is formulated as both a mixed integer programming problem and a stochastic programming problem. Since the model has binary variables, it is a MIP, as described in section 4.2.1. Since the model is stochastic, there are two stages of decision variables: first stage and second stage, which is explained in section 4.2.2. The first stage decision variables describe the decisions that will be made today. In this case, the only first stage decision variable is x_d. This means that today the model will choose the number of days and the specific days that the vessels will be offshore without knowing for sure what the weather situation will be those days. This must happen before knowing the weather situation to prepare the vessels, personnel, and tools. The second stage decision variables describe the decisions that will have to be made tomorrow. In this case, the second stage decision variables are y_{dws} and z_{dws}. In the second stage, the weather is known, making it possible to decide the number of hours that should be spent on maintenance every day.

6.3.1 Objective function

The result of the problem formulation step from section 4.1 concerning to *specify the objective function* is presented in this section. The objective is to minimize the number of days that the maintenance vessel and teams stay offshore. This is a single objective optimization problem, meaning that there is only one decision variable that is being optimized, as seen in equation 6.2.

$$min \sum_{d \in D} x_d \tag{6.2}$$

6.3.2 Constraints

The result of the problem formulation step from section 4.1 concerning to *specify constraints* is presented in this section. Constraints on the operational decisions are expressed in terms of the following inequalities. The total daily maintenance workload cannot exceed the maximum workload of the maintenance teams all together, which is described in constraint 6.3.

$$\sum_{w \in W} y_{dws} \leq C \cdot N \quad d \in D, s \in S \tag{6.3}$$

The number of available maintenance hours on a single turbine cannot exceed the maximum daily workload of a single maintenance team, which is considered through constraint 6.4.

$$y_{dws} \leq C \quad d \in D, w \in W, s \in S \tag{6.4}$$

On offshore days, constraint 6.5 limits the total available workload to not exceed the number of available hours those days.

$$\sum_{w \in W} y_{dws} \le A_{ds} \cdot x_d \quad d \in D, s \in S \tag{6.5}$$

Constraint 6.6 makes sure that the maintenance is done in a specific order.

$$z_{dws} \ge z_{d(w+1)s} \quad d \in D, w \in \{w_1, ..., w_{n-1}\}, s \in S \tag{6.6}$$

Constraint 6.7 ensures that every wind turbine gets its required number of maintenance hours that year.

$$z_{(d=365)ws} = T \quad w \in W, s \in S \tag{6.7}$$

Constraint 6.8 keeps track of the number of maintenance hours that have been performed on each turbine up until a given day.

$$z_{dws} = \sum_{i=0}^{d+1} y_{iws} \quad d \in D, w \in W, s \in S \tag{6.8}$$

Constraints 6.9 and 6.10 ensure that maintenance on turbine w is finished before maintenance on turbine $w+4$ is started. This means that maintenance on one turbine must be finished before a maintenance team can move on to doing maintenance on another wind turbine. Since there are four maintenance teams that are equally as efficient as each other, one team can start doing maintenance on the $(w+4)^{th}$ turbine when they finish doing maintenance on the w^{th} turbine. These constraints have been reformulated from the following If-Then statement: If $y_{d(w+4)} > 0$, Then $z_{dw} \ge T$.

$$T - z_{dws} \le M \cdot q_{dw} \quad d \in D, w \in \{w_1, ..., w_{n-4}\}, s \in S \tag{6.9}$$

$$y_{d(w+4)s} \le M \cdot (1 - q_{dw}) \quad d \in D, w \in \{w_1, ..., w_{n-4}\}, s \in S \tag{6.10}$$

Constraints 6.11 and 6.12 ensure that the vessel stays offshore 14 consecutive days. These constraints have been reformulated from this If-Then statement: If $x_d - x_{d-1} > 0$, Then $\sum_{i=d+1}^{d+13} x_i \ge 13$. For these constraints to work, constraint 6.13 must be included. If $x_{d=0} = 1$, constraint 6.12 would not be able to record that the first day of this year was the first day (out of the 14 consecutive days) offshore, since it does not have the last day of the previous year to compare it to.

$$13 - \sum_{i=d+1}^{d+13} x_i \le M \cdot p_d \quad d \in D \tag{6.11}$$

$$x_d - x_{d-1} \leq M \cdot (1 - p_d) \quad d \in D \tag{6.12}$$

$$x_{d=0} = 0 \tag{6.13}$$

Constraints 6.14 and 6.15 ensure that the vessel stays onshore for at least 28 consecutive days after the 14 days offshore. The 28 days onshore represents the four weeks when the vessel and maintenance teams have to perform maintenance on Dogger Bank B and Dogger Bank C. These constraints have been reformulated from the following If-Then statement: If $x_d - x_{d-1} > 0$, Then $\sum_{i=d+14}^{d+41} x_i \leq 0$.

$$\sum_{i=d+14}^{d+41} x_i \leq M \cdot h_d \quad d \in D \tag{6.14}$$

$$x_d - x_{d-1} \leq M \cdot (1 - h_d) \quad d \in D \tag{6.15}$$

One of the problem formulation steps from section 4.1 is *modeling*. This step has been completed by pooling together the specification of parameters, variables, objective function, and constraints in a consistent mathematical model.

6.4 Results

The optimization model from the previous section was implemented in Python, and the code can be found in Appendix B.1. The following assumptions leading up to the results in Table 6.3 have been made: there are four available maintenance teams on the mother craft vessel, each wind turbine in the studied wind farm requires 20 hours of maintenance per year, and each maintenance team can work for a maximum of 13 hours per day.

The second last problem formulation step from section 4.1 is to *run the model* which is executed in this section. Table 6.3 shows the results of running the optimization model over one year with 96 wind turbines. As the table shows, it took over two weeks to find the first solution / objective value of 42 days with an optimality gap of 12%. To decrease the running time, the model was divided into three sub-models. The first sub-model is run over the first four months of a year with the first third of the wind turbines. As Table 6.3 shows, it took less than 11 minutes to find an optimal solution (with an optimality gap of 0%) of 14 days offshore. The second sub-model is run from May until August with the second third of the wind turbines. It took just over 47 minutes to find an optimal solution of 14 days offshore. The final sub-model is run over the last four months with the final third of the wind turbines. The optimal solution of 14 days offshore for this sub-model was obtained after just 34 minutes. The sum of the three optimal solutions found in the three sub-models add up to be equal to the solution found by the entire optimization model. However, the time it takes to find the optimums from the three sub-models is significantly shorter than the time it takes to find the global optimum from the optimization model. Both solution methods resulted in the same objective value. Therefore, it seems smart to divide this model into sub-models to get a shorter total run time. However, the resulting schedules found in the sub-models were not the exact same as the resulting schedule found in the optimization model, meaning that this solution didn't necessarily find the global optimum or that there might be multiple optimal solutions.

Table 6.3: Results from running the optimization model and the three sub-optimization models

	Optimization model	*Sub-model 1*	*Sub-model 2*	*Sub-model 3*
Number of days used in model	365	120	123	122
Months used in model	Every month in a year	January - April	May - August	September - December
Turbines numbers used in model	All turbines	1 - 32	33 - 64	65 - 96
Time to find solution	1277884s	652s	2833s	2047s
Processor	2x 2.4GHz Intel Xeon Gold 5115 CPU – 10 core	2.9 GHz Dual-Core Intel Core i7	2.9 GHz Dual-Core Intel Core i7	2.9 GHz Dual-Core Intel Core i7
Objective value (best solution found)	42	14	14	14
Offshore days found in solution (day of the year)	-	April 15th (day 105) - April 28th (day 118)	July 17th (day 198) - July 30th (day 211)	October 26th (day 299) - November 8th (day 312)
Optimality gap	12.0%	0.0%	0.0%	0.0%

6.5 Discussion

6.5.1 Comparison to models from literature

The authors reviewed in Chapter 5, Dai et al. (Dai et al., 2015), Stålhane et al. (Stålhane et al., 2015), Gundegjerde et al. (Gundegjerde et al., 2015), Stålhane et al. (Stålhane et al., 2016), Gutierrez-Alcoba et al. (Gutierrez-Alcoba et al., 2017), and Irawan et al. (Irawan et al., 2017), have set the objective function in their models to minimize the total costs. Additionally, they have all incorporated some sort of penalty cost in their models. These authors have specialized within the field of optimization, and their proposed models are therefore much more complicated than the model proposed in this report. It would be possible to minimize the cost of days offshore in the objective function in the model proposed in this report, but it would not have made a difference since the model is only a single objective optimization model. This means that the decision variable being optimized does not have to be weighted, in terms of costs, against any other decision variables.

The model proposed in this report differs from the model proposed by Dai et al. (Dai et al., 2015) in that this model allows the service vessels to stay offshore multiple days, while their model forces it to return to the harbor every day. Some similarities between the model proposed in this report and the model proposed by Dai et al. are that they both restrict the number of working hours per day. The model proposed in this report restricts the number of hours each team can work, while their model restricts how many hours the vessel can be operated. The model proposed by Dai et al. does not consider any major replacements or repairs which is the same assumption in this model. Their model considers every wind turbine to represents one maintenance task, while the model proposed in this report considers that every turbine needs several hours per year. In addition, both models restrict the value of the decision variables that say if the vessel is offshore or not to be binary.

The arc-flow model proposed by Stålhane et al. (Stålhane et al., 2015) limits maintenance tasks to only be performed once, while this report's model considers that every turbine needs a number of hours per year. Both models consider a continuous vessel route between the turbines in the farm. The arc-flow model proposed by Stålhane et al. keeps track of the technicians on board a vessel and what times the different turbines are visited. This report's model, however, simplifies this aspect in the pre-processed data by subtracting a certain amount of time for travel from the given available hours to perform maintenance per day given stochastic weather. In addition, the model proposed in this report considers that there are several available maintenance hours that can be performed per day, but this model does not keep track of the individual technicians like the arc-flow model of Stålhane et al. does. Both models have a binary decision variable where their variable describes if a vessel travels between two turbines and the model proposed in this report describes if the vessel is operating offshore or not.

The model proposed by Gundegjerde et al. (Gundegjerde et al., 2015) included many advanced constraints for its stage 1 and stage 2 which were too advanced to be incorporated in the model proposed in this report. In stage 3 there are certain constraints that are similar to the constraints of this report's model. The safety constraints from the model proposed by Gundegjerde et al. that make sure that maintenance is not performed in rough weather conditions have been simplified in this report's model. This factor is considered in the pre-processed weather data of the model proposed in this report through wave height limits and wind speed limits.

Like the model proposed by Stålhane et al. (Stålhane et al., 2016), the model proposed in this report considers uncertainty in weather conditions. The model proposed by Stålhane et al. can consider multiple years, whereas the simple model proposed in this report only considers one year. Additionally, their model can consider multiple wind farms, while the model proposed in this report only considers one wind farm.

Like the model of Gutierrez-Alcoba et al. (Gutierrez-Alcoba et al., 2017), the model proposed in this report ensures that the first maintenance activity to be started is first to be finished.

Similar to the model proposed by Irawan et al. (Irawan et al., 2017), the model proposed in this report considers the weather windows to find the number of available hours that can be worked per day. In the model proposed in this report, this information is treated in the pre-processed data for real data given by Equinor for three years. The model proposed by Irawan et al. (Irawan et al., 2017) has a big focus around the vessels. In the model proposed in this report, this information is also treated in the pre-processed data. The model proposed by Irawan et al. is more advanced and considers more aspects such as different O&M bases, multiple vessels and wind farms, different types of maintenance tasks, and different costs.

6.5.2 Limitations to the proposed model

Section 6.1.1 shows that it is significant to find a good path between the many turbines at Dogger Bank A. The turbines that are furthest apart are 37.0 km away from each other. A normal vessel travels about 13 kn ≈ 24.1 km/h, meaning that it would take more than 1.5 hours to travel this distance. However, section 6.2 demonstrates that finding the shortest path between the many turbines is quite difficult to do with a computer. There are many combinations to check, and it would require a strong computer to be able to check all of them. Further, the global best route for performing maintenance activities at the turbines might not be the one with the shortest total travel distance. First, the SOV should probably travel back to the dock multiple times before it has visited all of the turbines. Secondly, the turbines should to some degree be visited based on their condition. Perhaps the turbines could be divided into smaller clusters of turbines that have similar conditions and then that route could be optimized. Another important factor to consider when scheduling a good maintenance route for the turbines is the uncertain weather. Weather parameters such as wave height and wind speed are stochastic parameters, and it is important to treat them that way and plan to do maintenance relatively early.

The optimization model that has been proposed in section 6.3 is rather simple. It only considers a decision vector but not a cost vector in the objective function, as suggested in section 4.2.1. As section 6.5.1 shows, many other optimization models from the literature aim to minimize the total cost. It could have been beneficial to have costs tied to the decision variable in the proposed model as well. This way the model could have been more flexible and taken more aspects into consideration such as penalty costs for overtime work. If a turbine was about to fail and would not have continued operating until the next trip, it might be more cost beneficial to work overtime and extend the vessel rent contract instead of letting the turbine fail and then fix it the next time. Furthermore, the proposed model returns the days, turbines, and the number of hours spent on maintenance on the different turbines each day. From the results in Table 6.3, the model only suggests one two-week period per sub-model. However, there were probably more periods which would have been almost as good. For a future optimization model for routing and scheduling, it could be beneficial to generate multiple sets of optimal solutions or close to optimal solutions. This way the condition of the turbines could have a bigger influence on when maintenance should be performed. Additionally, the model could have had more soft constraints for a more flexible and perhaps better solution. Constraints 6.11 and 6.12 force the vessel to stay offshore for 14 consecutive days. However, if the weather was worse than expected the last two days of the offshore period at Dogger Bank A, it might be better to return the vessel to the dock instead of waiting offshore without performing any maintenance. The proposed model only considers one of the tree offshore wind farms at Dogger Bank. A model for the future should consider all three wind farms instead of just forcing the model to stay away from Dogger Bank A for a minimum of 28 days through constraints 6.14 and 6.15.

Chapter 7

Optimization and predictive maintenance

This chapter starts by giving a summary of the PdM model used in the case study of Lien (Lien, 2021). It goes on to present a catalog of different approaches that can be used to combine optimization models and PdM models.

7.1 Predictive maintenance model

This section gives a brief recap of the different formulas and methods used in the case study of Lien (Lien, 2021) that are reused in the case study in Chapter 8. A more detailed summary of the model can be found in Appendix C.

7.1.1 Degradation model

This section is based on the work of Rausand et al. (Rausand et al., 2021). Degradation models are used for condition-based maintenance decisions for an item experiencing an observable degradation. One can distinguish between:

- $X_t, t \geq 0$: a stochastic process describing the actual degradation of the item at time t.
- $Y_t, t \geq 0$: a stochastic process describing the degradation measurements of the item at t.

The measurements can contain errors, such as noise, which the Wiener process with linear drift incorporates. Examples of degradation are vibration levels, crack lengths, and corrosion depths.

For the sake of simplicity, perfect measurements of the degradation are assumed, i.e., $X_t = Y_t$ for the case study. In this chapter, X_t-notation is used, while in the case study Y_t-notation is used.

7.1.2 Wiener process with linear drift

From section 3.4, the Wiener process is a statistical data-driven approach. Furthermore, the Wiener process is often considered for “direct condition monitoring data, which is described in section 3.4.2. This rest of this section is based of Rausand et al. (Rausand et al., 2021). The Wiener process, $W_t, t \geq 0$ is defined as follows:

1. **Initial condition** $W_0 = 0$
2. **Independent increments** W has independent increments, meaning that for every $t > 0$, the future increments $W_{t+u} - W_{t,u} \geq 0$, are independent of the past values $W_s, s \leq t$.
3. **Normal increments** W has Gaussian increments, meaning that $W_{t+u} - W_t$ is normally distributed with mean 0 and variance u: $W_{t+u} - W_t \sim \mathcal{N}(0, u)$.
4. **Continuous paths** W has continuous paths, meaning that W_t is continuous in t.

Similar to the Wiener process, the Wiener process with linear drift holds the following properties:

1. **Initial condition** $X_0 = x$
2. **Independent increments** X has independent increments, meaning that for every $t > 0$, the future increments $X_{t+u} - X_{t,u} \geq 0$, are independent of the past values $X_s, s \leq t$.
3. **Normal increments** X has Gaussian increments, meaning that $X_{t+u} - X_t$ is normally distributed with mean 0 and variance u, hence $X_{t+u} - X_t \sim \mathcal{N}(\mu u, \sigma^2 u)$.
4. **Continuous paths** W has continuous paths, meaning that W_t is continuous in t.

The Wiener process with linear drift μ and infinitesimal variance σ^2 can define the following stochastic process:

$$X_t = x + \mu t + \sigma W_t, \tag{7.1}$$

where X_t is normally distributed with mean μt and variance $\sigma^2 t$. The theory of stochastic processes makes it known that when the process reaches level l for the first time, the time T is inverse-Gauss (IG) distributed with parameters $\alpha = l/\mu$ and $\beta = (l/\sigma)^2$. For the inverse-Gauss distribution, $X \sim \text{IG}(\alpha, \beta)$, the probability density function (PDF) is described in equation 7.2 and the cumulative distribution function (CDF) is described in equation 7.3.

$$f_X(x; \alpha; \beta) = \sqrt{\frac{\beta}{2\pi x^3}} \exp(-\frac{\beta(\text{x} - \alpha)}{2\alpha^2 \text{x}}), \tag{7.2}$$

$$F_X(x;\alpha;\beta) = \phi(\frac{\beta}{\alpha}\sqrt{x} - \sqrt{\beta}\frac{1}{\sqrt{x}}) + \phi(-\frac{\beta}{\alpha}\sqrt{x} - \sqrt{\beta}\frac{1}{\sqrt{x}})e^{\frac{2\beta}{\alpha}} \tag{7.3}$$

Furthermore, for $\Delta X = X_{t+\Delta t} - X_t \sim \sqrt{\Delta t}\mathcal{N}(\mu,\sigma^2)$, the expected value and variance are given by:

$$\mathrm{E}(\Delta X) = \mu\Delta t, \mathrm{Var}(\Delta X) = \sigma^2\Delta t \tag{7.4}$$

7.1.3 Cost function

Consider that the degradation process is continuously observed without uncertainty. A failure occurs if $X_t \geq l$ for some time t. A request to replace the component is placed when degradation approaches the failure limit l. The objective is to determine the maintenance limit $m < l$. The maintenance limit is assumed to already have been reached at the starting point, and the RUL should be estimated. RUL is a stochastic process and a distribution that covers all of the degradation trajectories. From section 3.2.1, this means that confidence and uncertainty intervals related to the estimation can be achieved. Furthermore, this model only considers one maintenance cycle. $X_t = m$ is assumed to be observed at time t in this cycle. RUL_m is the time from t until a failure occurs, and it is inverse-Gauss distributed with parameters $\alpha = (l-m)/\mu$ and $\beta = (l-m)^2/\sigma^2$, given that our starting point is the maintenance limit m (Rausand et al., 2021). The following cost function, from Vatn (Vatn, 2020a), is used to minimize the maintenance limit m:

$$C(m) = \frac{c_{\mathrm{R}} + c_{\mathrm{F}}F(T_{\mathrm{L}}|m) + c_{\mathrm{U}}\int_0^{T_{\mathrm{L}}} f(t|m)(T_{\mathrm{L}} - t)dt}{\mathrm{MTBR}(m)}, \tag{7.5}$$

where c_{R} is the cost of renewal or replacement. c_{F} denotes the cost of failure. c_{U} is the cost per hour of down time. $F()$ and $f()$ are respectively the CDF and the PDF for RUL, given that we are at the maintenance limit m at some point. Finally, $\mathrm{MTBR}(m)$ is the Mean Time Between Renewals, given the decision rule to request a maintenance at m.

7.1.4 Results

Lead times affect the maintenance costs, and the time maintenance should be performed. Long lead times lead to short maintenance limits and high costs, while short lead times lead to long maintenance limits and low costs. (Lien, 2021)

7.2 Catalog: Combining optimization and PdM models

There are multiple approaches that can be tackled to link optimization modeling for routing and scheduling with the PdM modeling. The following is a catalog with different possible approaches that can be or have been explored. The approaches are structured into a *problem statement* that describes the optimization problem, *decision variables* that describe the quantities that should be determined, and *modeling challenges* tied to implementing the approach.

7.2.1 Approach 1

Problem statement

This approach focuses on the planning perspective and aims to make to most cost-beneficial decision. Assume that a maintenance schedule is already found and that the planned trips are already known. Consider that the maintenance teams get alerted while offshore, that turbine X, which is not scheduled for maintenance on this trip, has reached the maintenance limit and therefore needs maintenance. This approach only considers turbine X, which has reached the maintenance limit. Now, there are three different options for how to proceed with planning maintenance for turbine X:

1. Wait until the next trip and assume that you are then flexible and can plan quite freely, so that the extra cost of getting turbine X is almost equal to zero.
2. Include it in the current trip, but at a cost of having to reschedule some of the planned turbines to the next trip, with an extra cost per turbine that you must reschedule.
3. Choose another route that is closer now, meaning that you must change two routes, where there is a cost of changing between e.g. «route A» and «route B».

Decision variables

It could be an idea to have a binary variable that says if maintenance on turbine X is included in this trip, option 2 or 3, or if it is waited with until the next trip, option 1. This decision variable could look like equation 7.6.

$$X = \begin{cases} 1, & \text{if turbine X is moved earlier} \\ 0, & \text{if turbine X is waited with} \end{cases} \tag{7.6}$$

If turbine X is waited with, the risk of the turbine failing must be calculated, and the moving cost will be equal to zero. The PdM-model in equation 7.5 could be used to calculate the expected cost of waiting to perform maintenance on turbine X until the next trip. If turbine X is moved earlier, the risk of failing will be equal to zero, but there will be a moving cost of either single turbines for option 2, or entire routes for option 3. Then, if $X = 1$, there could be another binary variable that says whether individual turbines must be rescheduled or if the entire route must be rescheduled. This decision variable could look like equation 7.7.

$$Y = \begin{cases} 1, & \text{if individual turbines are rescheduled} \\ 0, & \text{if the entire route is rescheduled} \end{cases} \tag{7.7}$$

If $Y = 1$, individual turbines are rescheduled, and there must be a moving cost for each individual turbine. Furthermore, an expected cost should be calculated for the individual turbines that are being moved. Otherwise, if $Y = 0$, there must be a cost of changing two routes, and the expected cost of the turbines should be considered.

Modeling Challenges

- A potential challenge could be to model the decision about switching two routes. Then the turbines that are moved earlier might get maintenance too early which costs more, and some turbines in the other route will have a higher risk of failing.
- Solving an optimization model that can consider the expected cost of all potential moves could end up having a very long run time.
- Finding a good way to assign costs and consider the most important aspects involved.

7.2.2 Approach 2

Problem statement

The case study in the specialization project (Lien, 2021) aims to find the most cost-beneficial maintenance limit for one single turbine under uncertain lead times. For this approach, it is important to be able to consider multiple turbines and make a cost-beneficial schedule for the entire wind farm. Even though it might be more cost-beneficial for a single turbine to await maintenance, it might not be the most cost-beneficial for the entire wind farm which must consider a daily capacity and traveling time between turbines. Furthermore, the uncertainty in the lead time can now be reduced because the optimization model for routing and scheduling gives us control over planning which can result in greater gains. For this approach, the schedules must be updated daily based on the real condition of the individual turbines. Now, the main questions are:

1. What are the conditions of the turbines that are scheduled for maintenance today and can they be postponed, e.g. is there enough capacity in the next maintenance window and will the turbine fail before then?
2. Are there any turbines that require maintenance earlier than they were scheduled, and are there other "healthier" turbines scheduled earlier that can be rescheduled for later?

Decision variables

The decision context is that when a trip starts, the condition of each turbine is known. This information can be used to prioritize which ones will be maintained on the current day, and the order of when they will get maintenance. This approach is further studied in Chapter 8.

Modeling challenges

- One of the main challenges with this approach is avoiding that all of the turbines are postponed until the last day of the maintenance window.
- More challenges related to this approach are discussed in Chapter 8.

7.2.3 Approach 3

Problem statement

Assume that several maintenance windows with a set length have been given. If the condition of some turbines scheduled for the next window started degrading quickly towards the end of this maintenance window, this approach allows for extending the maintenance window with extra days. There is a chance that the turbines will make it until their scheduled window. However, the expected cost of getting there is quite high. Now there are two different options for how to proceed with planning the maintenance:

1. Stick to the original length of the given maintenance windows and maintain the most impaired turbines first in the next window while being prepared to pay the expected cost of waiting.

2. Allow for changes of the length of the maintenance windows so that this maintenance window can be extended with the extra cost of extending the rental contract of the SOV.

Decision variables

In this approach, the decision to make is quite simple: The maintenance window will be extended if the expected cost of waiting is higher than the cost of extending the rental contract of the SOV. If the maintenance window is not extended, there is a risk that the fast-degrading turbines will fail, but there will not be any extra costs related to extending the rental contract of the SOV. The risk can be calculated by using the PdM-model in equation 7.5. If the maintenance window is extended, there will be no risk of turbine failure, but there will be costs tied to extending the rental contract. The decision variable can be expressed as equation 7.8.

$$X = \begin{cases} 1, & \text{if the length of the original maintenance window is kept} \\ 0, & \text{if the length of the original maintenance window is extended} \end{cases} \tag{7.8}$$

Modeling challenges

- If one maintenance window is extended with an extra cost, the next window could be shortened since there will be fewer turbines to maintain. It is reasonable to assume that this would be cheaper, and this aspect should also be included in the model.
- Calculating the cost of changing the length of the maintenance windows, both to be longer and to be shorter.

7.2.4 Approach 4

Problem statement

Assume that maintenance must be performed on a given number of turbines within a year. Having fewer maintenance vessels and maintenance teams will require more frequent visits to complete maintenance for all of the turbines. The benefit of this is that there will be more maintenance windows within which to perform maintenance. If the numbers of maintenance vessels and teams are increased, the overall maintenance could be finished in a shorter amount of time and require fewer maintenance windows. Furthermore, the maintenance limit can be "stretched" further for more turbines since the daily capacity will increase. This will lead to reduced costs due to maintenance being performed closer to the most "ideal" time. However, the cost of renting more assets would increase the overall costs. The goal is to find the most cost-beneficial solution. Here are the options:

1. Stick to the original maintenance plan and use one SOV to maintain all of the turbines throughout the year. Furthermore, this option accepts that some turbines will get maintained much earlier than required to get through all turbines while reducing the risk of failure in case of delays.

2. Rent more DCs and maintenance teams so that the maintenance limit can be “stretched” and perhaps reduce the number of trips.
3. Just hire more maintenance teams and not more DCs, to get more maintenance done per trip.

Decision variables

One of the decisions that should be kept track of in this approach is the number of DCs that are rented. The decision variable could be denoted as:

$$X = \text{The number of DCs that are rented} \tag{7.9}$$

The main benefit of having more DCs is that they give more flexibility when it comes to transporting the maintenance teams out to the wind turbines. However, it comes with a high rental cost. Then, the next decision variable should keep track of the number of maintenance teams. The decision variable could be denoted as:

$$Y = \text{The number of maintenance teams that are used} \tag{7.10}$$

The main benefit of having more maintenance teams is that more turbines could be maintained at the same time. Furthermore, the maintenance limit could be "stretched" longer since the daily maintenance capacity would be bigger. This could save some costs because then maintenance would be done closer to the ideal time, and the turbines would have to be maintained fewer times. However, a greater risk is taken by stretching the maintenance limit since there are more uncertainties involved now. For instance, a DC might not work one day, the wave height might get too great for the DCs to operate, or one maintenance team might get the flu and not be able to work. Furthermore, if the turbines are maintained fewer times, this implies that there will be fewer trips to the wind farm and thereby fewer maintenance windows to place the wind turbines in. This means that they would not all get maintenance on the ideal time. Additionally, having more maintenance teams costs more, and could potentially get inefficient if there are too many teams. For instance, several of the maintenance teams might be forced to wait longer on the SOV if the DCs cannot be used one day.

Modeling challenges

- The main challenge with modeling this approach is that the optimization model will become much more complicated if more DCs and maintenance teams are considered, because there will be more logistics such as routes to keep track of. Moreover, if the model becomes very complex, it can take a long time to solve.

Chapter 8

Case study

For the case study, *Approach 2* from the catalog in section 7.2 was chosen to go ahead with. The original idea was to test out more approaches from the catalog, but it took more time to create a case study for *Approach 2* than expected. It was challenging to create a maintenance model that updates the schedule daily based on the condition of each turbine and return a feasible schedule. The risk is that too many turbines might be pushed to the last day of the maintenance period and exceed the capacity. This chapter first presents *Approach 2* in section 8.1 to describe how maintenance activities are scheduled and updated in time. Furthermore, four heuristics are proposed in section 8.2 which force a minimum number of turbines to be maintained within a given time frame to reduce the risk of the daily capacity being exceeded some days.

8.1 Approach 2

8.1.1 Generating the degradation paths

In this case study, the Wiener process with linear drift, is used for degradation modeling, with degradation indicator Y. This case study uses the results of sub-models 1 in Table 6.3. It is assumed that there are $q = 32$ identical and independent turbines. Let t denote the running day during the planning horizon from $t_0 = 104$ up to $T = 118$. The first possible day to carry out maintenance is $t = t_0 + 1 = 105$. Let $Y_{i,t}$ denote the observed condition of turbine i at day t. This case study sets $Y_{i,t_0} = 88$ for all turbines, i.e., they are getting close to their maintenance limits. In is straight forward to generalize the initial condition by having a distribution over the Y_{i,t_0}-values. Monte Carlo simulation is used to simulate the degradation where the increments are pseudo random number from the normal distribution with $\mu = 1$ and $\sigma = 1$. There are 72 observations per day, making the total number of observations within the time frame of two weeks more than 1000. The degradation paths for all $q = 32$ units are illustrated in Figure 8.1.

The turbines have reached their failure limit l when the condition $Y_{i,t} = 100$, which is illustrated as a red line in Figure 8.1. The blue line indicates an approximate maintenance limit m, meaning the maintenance limit is not fixed. When the condition is $Y_{i,t} = 96$ or more on a given day, the cost equation 7.5 shows that the expected cost is lowest if maintenance is performed on that day, i.e., the most cost-beneficial lead time $T_L = 0$. By waiting longer, the maintenance costs get lower, but the risk of failing gets higher. However, a turbine could be scheduled for maintenance before the condition has reached $Y_{i,t} = 96$ because the condition is rounded to the nearest integer. Furthermore, if $t = T$ a turbine is scheduled for maintenance regardless of the condition because it is too risky to wait until the next planned maintenance period.

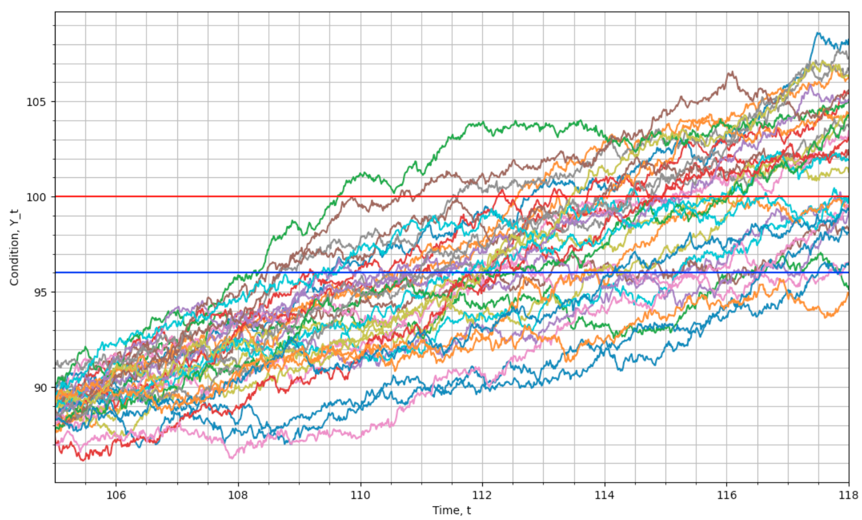


Figure 8.1: Example of degradation paths for 32 turbines in the optimal maintenance period from sub-model 1, Table 6.3

8.1.2 Optimization problem

The challenge is to plan which turbine should be maintained at each day $t = t_{0+1},...,T$. Now, assume that on day t it is being considered which day to schedule maintenance for turbine i. Let $x_{i,t}$ denote the number of days after t maintenance should be planned (to be performed) for this turbine, where $x_{i,t} = 0$ means that maintenance is planned for day t.

In the planning process it is assumed that if $x_{i,t}$ is planned with for some fixed day, we may, with reference to the PdM model from section 7.1, let $x_{i,t}$ represent the lead time, i.e., T_L. Given $T_L =$

$x_{i,t}$ and considering the current state $Y_{i,t}$ to represent the maintenance limit, m, the expected cost of the strategy could be calculated. Let $T_{\mathrm{L}} = x_{i,t}$ and $m = Y_{i,t}$ and from the cost function in Equation 7.5 follows:

$$c(m) = c(m = Y_{i,t}, T_{\mathrm{L}} = x_{i,t}) \tag{8.1}$$

Computationally it is convenient to store calculated values for the function $c(m, T_{\mathrm{L}})$ for fast look-up. Figure 8.2 depicts the cost function $c(m, T_{\mathrm{L}})$ as a function of the maintenance limit for various lead times, whereas Figure D.1 (in Appendix) depicts $c(m, T_{\mathrm{L}})$ as a function of the lead time for various maintenance limits. The figure clearly shows that long lead times involve a bigger risk of failing.

Let $\mathbf{x}_t = [x_{1,t}, ..., x_{N,t}]$ be the vector of due dates when planning at day t. The optimization problem is now to minimize:

$$Z = Z(\mathbf{x}_t) = \sum_i c(Y_{i,t}, x_{i,t}) \tag{8.2}$$

The calculations in equation 8.2 are then continuously, i.e., for each day, updated. First, a task is placed out in time to minimize equation 8.2. On day t, $x_{i,t}$, which represents the lead time, is decided. This is a rather conservative approach because when the next day comes the lead time can be reduced, if necessary, i.e., if the degradation develops faster than initially anticipated.

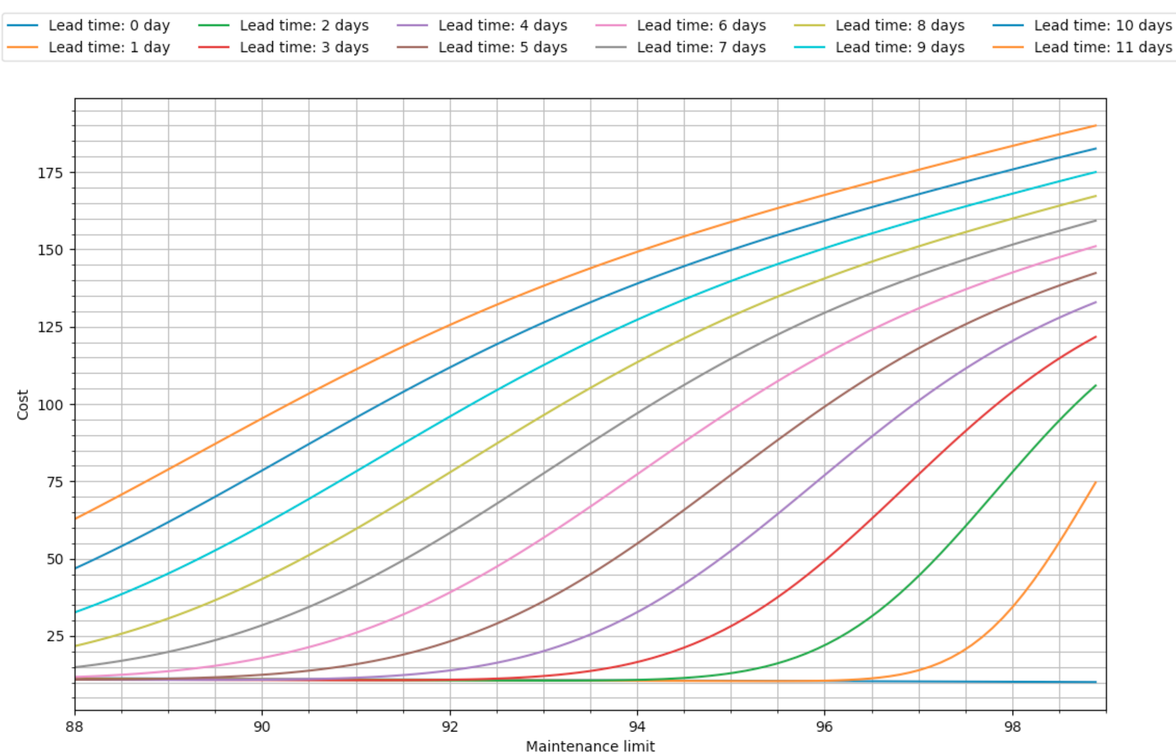


Figure 8.2: Expected costs of maintenance limits with varying lead times

8.1.3 Explanation of algorithm

This model updates the maintenance schedule every day t based on $Y_{i,t}$. The proposed algorithm can be divided into three main parts, namely *initial calculations*, *create initial schedule*, and *update schedule*. A simple illustration of the algorithm is presented in Figure 8.3.

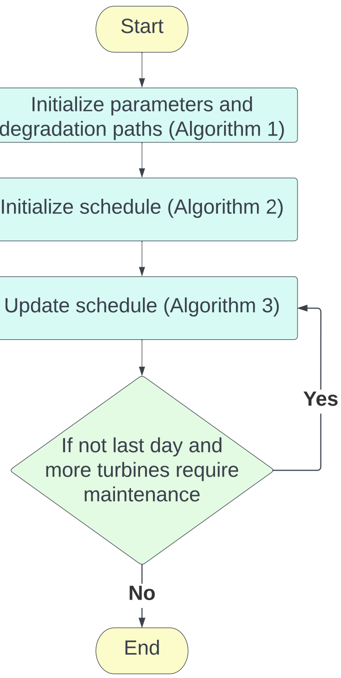


Figure 8.3: Flowchart of Approach 2.

For a better understanding of the algorithm, a practical step-by-step example of running the code is provided in Appendix D.2. Furthermore, the code written for this algorithm can be found in Appendix B.2. The main variables are the maintenance limits, $Y_{i,t}$, and the lead times, $x_{i,t}$. All three parts are presented in the following:

Initial calculations. This part of the algorithm covers calculations that are made initially such as simulating the degradation trajectories for 32 turbines and calculating different cost and condition tables. This is expressed in Algorithm 1.

Algorithm 1 Pseudo code for the implementation of algorithm for *Initial calculations*.

Generate degradation paths for q turbines with $72 * 14$ steps;
Store the daily conditions $Y_{i,t}$ of the turbines;
Set up a function for cost equation 7.5;
Calculate and store values $(\mathbf{x}^*_{i,t}, \mathbf{Y}_{i,t})$ from equation 8.1 for fast look-up;

Create initial schedule. This part creates an initial maintenance schedule for the first day of the maintenance period, $t = 105$. First, the turbines are sorted based on their condition, $Y = i, t = 105$. Thereafter, the most cost-beneficial lead times, $\mathbf{x}^*_{i,t}$, when planning for day t are found based on the conditions $Y_{i,t}$. Finally, the model uses this information to create an initial condition-based schedule on $t = 105$. This is expressed in Algorithm 2.

Algorithm 2 Pseudo code for the implementation of algorithm for *Create initial schedule*.

  Sort turbines from worst to best condition;
  for Every turbine and their condition **do**
    Retrieve $x^*_{i,t}$ from Algorithm 1 based on the condition;
    Place the turbine accordingly into a schedule;
  return Initial maintenance schedule;

Update schedule. The final part of the algorithm updates the schedule every day t, based on the daily conditions $Y_{i,t}$. If a turbine has gotten maintenance on a previous day, it is not considered for maintenance on a future day in this maintenance period. This is expressed in Algorithm 3.

Algorithm 3 Pseudo code for the implementation of algorithm for *Update schedule*.

  if All turbines have been maintained **then**
    return schedule, current day t, and maintained turbines;
  else
    Sort turbines from worst to best condition based on t;
    for Every not maintained turbine and their condition **do**
      Retrieve $x^*_{i,t}$ from Algorithm 1 based on the condition;
      if $x^*_{i,t} = 0$ or $t = T$ **then**
        Set ideal maintenance day to t;
      if $x^*_{i,t}$ takes turbine outside of maintenance period **then**
        Set ideal maintenance day to T;
      else
        Set ideal maintenance day according to $x^*_{i,t}$;
      Add turbine to the maintenance schedule based on the ideal maintenance day;
  return schedule, current day t, maintained turbines;

Algorithm 3 is repeated until all turbines have been maintained.

8.1.4 Results

Running the model with $q = 32$ turbines gives infeasible solutions since the daily capacity is exceeded some days in every test run. An example of running this model with $q = 32$ turbines can be found in Appendix D.3. For a simple demonstration of the results of running the model, the number of turbines is reduced to $q = 10$. The resulting schedule is a dictionary with *key=day* and *value=[list of scheduled turbines]*:

```
schedule: {105: [], 106: [], 107: [], 108: [3], 109: [], 110: [6], 111:
    [8, 9, 2, 1], 112: [5, 10], 113: [7], 114: [], 115: [4], 116: [], 117:
    [], 118: []}
```

The equivalent paths are illustrated in Figure 8.4. It is possible to check that the schedule makes sense based on the daily condition of a given turbine.

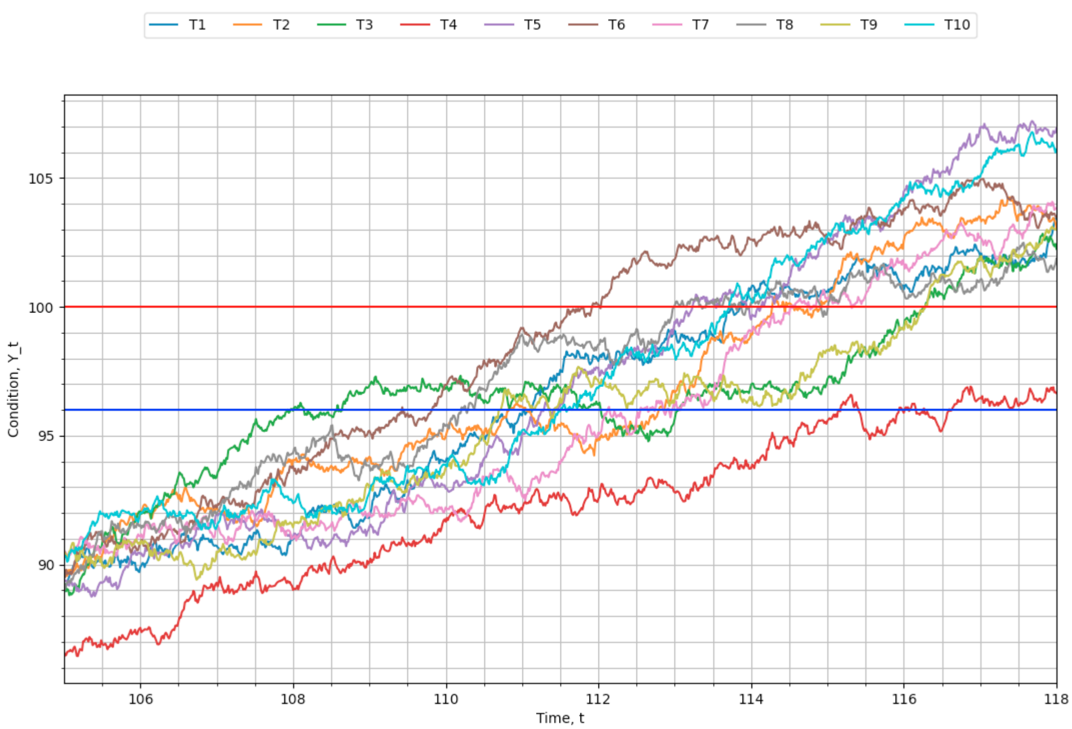


Figure 8.4: Degradation paths for 10 turbines, T#, in first period from sub-model 1, Table 6.3

The condition of the different turbines at the time they got maintenance is given in Table 8.1.

Table 8.1: The condition of the turbines at the time they got maintenance

Turbine	3	6	8	9	2	1	5	10	7	4
Condition	96.0	97.0	99.0	96.4	96.1	95.8	97.6	96.8	96.2	95.6

8.1.5 Discussion of model and alternative solutions

The proposed model is rather limited and does not account for every realistic scenario. The model comes with two main challenges: (1) There is a big chance that the capacity is exceeded some days. (2) The model is simplified and does not consider all stochastic parameters.

Exceeded capacity. When the last day of the planned maintenance trip comes, the turbines that have not yet been maintained must get maintenance, as it is too long to wait until the next trip. The risk is that some activities could be postponed until the last day or until a day with many other scheduled turbines and there would not be enough capacity to maintain all of them. This problem exists because the approach considers the turbines individually instead of as a part of a wind farm. Furthermore, if the capacity is exceeded one day, the capacity of a previous day might have been wasted.

Stochastic parameters. The model does not consider the weather as stochastic, but rather that the weather is good enough to do maintenance on every planned day. The risk is that the weather might change and some of the maintenance must be postponed, and the next windows might be full. Moreover, the model does not consider the mean (active) repair time (described in section 2.4) which to some degree is a stochastic parameter. Performing the maintenance might take longer than anticipated due to sickness of staff or missing equipment for example.

Solutions. Since there are only four maintenance teams, from section 1.1, the daily capacity limit could be four turbines. A naive approach would be to just impose this limit. However, if many turbines were scheduled to meet, but not exceed, the capacity limit the last days of the maintenance period, there would be no buffer. Realistically, this would lead to the capacity being exceeded some days because of stochastic parameters such as the degradation rate and the weather. On the other hand, if all turbines were scheduled and maintained at the first opportunity, the costs would be high because multiple turbines would get maintenance ahead of their ideal limit. It is clever to plan maintenance ahead to account for stochastic parameters and to reduce the risk of the capacity being exceeded, without being too conservative.

There should be a buffer planned for the turbines from the beginning for those that might need maintenance earlier than anticipated. The buffer could be a requirement of a minimum number of turbines that must get maintenance within a certain number of days. Furthermore, these first days could take more turbines if the turbines cannot wait longer given that the daily capacity is not exceeded. This solution schedules the turbines earlier to avoid delays at the end.

The PdM model from equation 7.5 could be used to calculate the expected cost and the turbines could be placed accordingly into an initial schedule. The schedule would consider the capacity limit and view each day as a placeholder. Then, the turbines would be placed within the placeholders depending on their condition, i.e., the turbines with the worst condition get maintenance first. The proposed algorithm uses the same initial condition for all turbines, leaving a small interval of the most cost-beneficial lead times. This makes it challenging to create an initial maintenance schedule since all of the turbines would be pushed to the beginning of the maintenance period to ensure that there would be enough time to maintain all of them. To avoid this problem, the initial conditions of the turbines could be distributed.

It could be beneficial to check the expected cost of turbines for the next days of the maintenance period. There would be certain criteria around moving turbines such as turbines scheduled the first or current day cannot be moved earlier, and turbines scheduled the last day cannot be moved later. The turbines scheduled on days in between could be moved both earlier and later if the requirements are not violated. Moreover, all turbines are allowed to hold their scheduled place if the capacity allows it. This procedure could be repeated daily based on the conditions of the turbines.

Some heuristics have been inspired by these ideas and are implemented and presented in section 8.2 to get some feasible solutions.

8.2 Heuristics for Approach 2

In these proposed heuristics the initial condition is distributed with the discrete uniform distribution between $Y_0 = 80$ and $Y_0 = 96$ for a more spread-out initial schedule. Furthermore, the planning horizon can now go from $t_0 = 105$ up to $T = 118$. It follows that there are 77 observations per day to make the total number of observations within the time frame more than 1000. There are two periods for every heuristic: *the first period* and the rest of the planning horizon. Let d denote the number of days in the first period. Furthermore, every heuristic has a criterion for the minimum number of turbines that must be maintained within the first period, n. Table 8.2 shows the two varying requirements for the four proposed heuristics.

Table 8.2: Requirements for the different heuristics

Heuristic	Number of days in the first period, d	Minimum number of turbines to maintain, n
H1	8	16
H2	7	10
H3	7	16
H4	7	14

It is important to emphasize that a heuristic here is a pragmatic way of solving a problem where one is not able to find good analytical ways to optimize, i.e., minimize the risk of exceeding the daily capacity the last days. By simulating many realizations, one can investigate which heuristic will be best in the long run.

The limitations of *Heuristic 1* are presented in Figure 8.5 as a visual example. The green marker indicates d, while n is presented below the marker.

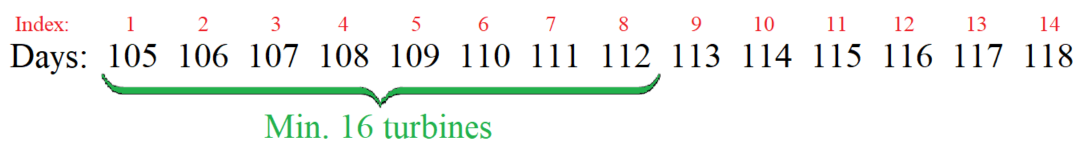


Figure 8.5: Example visualization of the first period, d, and the restrictions, n, (for *Heuristic 1*)

8.2.1 Performance example

The following are the result of running the heuristics. The model returns a schedule, on the form {day: [scheduled turbines]}, and a cost, which has been obtained from the cost equation 7.5:

```
H1 schedule: {105: [17], 106: [10], 107: [19, 4, 13, 30], 108: [12, 18],
    109: [16], 110: [3, 24, 27], 111: [25], 112: [29, 22, 7], 113: [5],
    114: [14, 6, 26], 115: [1], 116: [32, 8, 28], 117: [15, 21, 2, 31],
    118: [20, 9, 11, 23]}. Cost: 335.83761481532116
```

```
H2 schedule: {105: [17], 106: [10], 107: [19, 4, 13, 30], 108: [12, 18],
    109: [16], 110: [3, 24, 27], 111: [], 112: [25, 29], 113: [7], 114:
    [14, 6, 26], 115: [5, 1], 116: [32, 8, 22, 28], 117: [15, 21, 2, 31],
    118: [20, 9, 11, 23]}. Cost: 334.9513031857358
```

```
H3 schedule: {105: [17], 106: [10], 107: [19, 4, 13, 30], 108: [12, 18],
    109: [16], 110: [3, 24, 27], 111: [25, 7, 22, 8], 112: [], 113: [29],
    114: [14, 6, 26], 115: [5], 116: [32, 1, 28], 117: [15, 21, 2, 31],
    118: [20, 9, 11, 23]}. Cost: 335.59125714969497
```

```
H4 schedule: {105: [17], 106: [10], 107: [19, 4, 13, 30], 108: [12, 18],
    109: [16], 110: [3, 24, 27], 111: [25, 7], 112: [29, 22], 113: [5],
    114: [14, 6, 26], 115: [1], 116: [32, 8, 28], 117: [15, 21, 2, 31],
    118: [20, 9, 11, 23]}. Cost: 335.7781375790637
```

H1 schedule. The results show that there were 16 turbines scheduled for the first eight days, which meets the n criteria of this heuristic.

H2 schedule. The results show that there were 12 turbines scheduled for the first seven days, which meets the n criteria of this heuristic. In comparison to the test run of H1, the schedule is the same up until day 110. On day 111, H2 has already met the criteria and can leave this day open. H1 on the other hand must still perform maintenance on four more turbines before day 113 to meet the criteria. The resulting cost of the run in H2 is lower, indicating that some of the turbines scheduled from day 111 and later could have waited longer.

H3 schedule. The results show that there were 16 turbines scheduled for the first seven days, which meets the n criteria of this heuristic. The schedule of the results from running this heuristic is the same as the results of running H1 and H2 up until day 111. It is possible to see that this heuristic must perform maintenance on turbines 25, 7, 22, and 8 earlier to satisfy the given restrictions. However, this heuristic still performed better than H1, but worse than H2 on this run.

H4 schedule. The results show that there were 14 turbines scheduled for the first seven days, which meets the n criteria of this heuristic. This schedule is the same as all of the previous scheduled up until day 110. On day 111, this heuristic must perform maintenance on turbines 25 and 7 to satisfy the given restrictions, but unlike H3, H4 can wait maintaining turbines 22 and 8. Furthermore, turbine 8 is not scheduled until day 116 meaning that the degradation slowed down and that there were other turbines that needed maintenance before this one. However, for some reason this heuristic performed worse than H3.

8.2.2 Explanation of algorithm

This algorithm extends the algorithm created for *Approach 2* in section 8.1.3. The extensions of this algorithm can be divided into two main parts, namely *move turbine suggestion* and *move turbines and fix sequence of schedule.* A simple illustration of the algorithm is presented in Figure 8.6. Furthermore, the code written for this extension of the algorithm can be found in Appendix B.3. The two main parts are presented in the following:

Move turbine suggestion. This algorithm checks which of the scheduled turbines it is cheapest and possible to move on a given day. This function considers the current day, the current schedule, and d and n to prevent suggesting a move outside of the schedule or that violates the requirements. Finally, the algorithm suggests a turbine to move and a day to move it to. This is expressed in Algorithm 4.

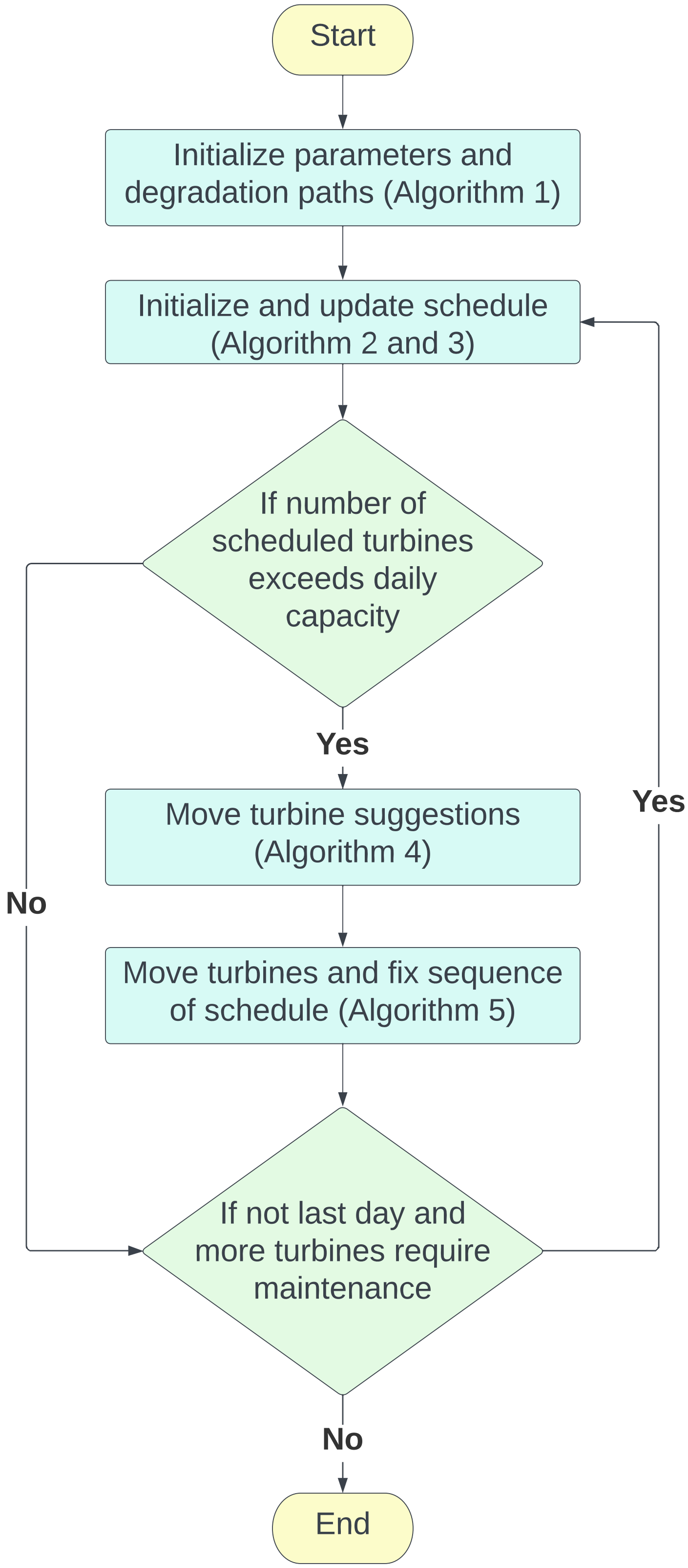


Figure 8.6: Flowchart of Heuristics.

Algorithm 4 Pseudo code for the implementation of algorithm for *Move turbine suggestion.*

```
Retrieve the sickest, i_s, and the healthiest turbine, i_h, from given schedule on given day, t;
if t = t_0 then
    i_h should be moved later to the first available day;
else if t = T or (t < t_0 + d and moving turbines later will not satisfy n) then
    i_s should be moved earlier to the first available day;
else
    Calculate the cost of moving i_s earlier and i_h later to the first available day;
    Calculate the price difference of moving the turbines from today;
return The cheapest possible turbine to move and the day to move it to;
```

Move turbines and fix sequence of schedule. This part first re-organizes turbines in the schedule to satisfy the given requirements: d and n, while not exceeding the capacity. For each day where the schedule exceeds the daily capacity, Algorithm 4 is used to find the most cost-beneficial suggestions on where to move turbines. It should be noted that this part only finds the best placeholders for the turbines, but all the turbines have not been moved according to their condition. Therefore, after all the requirements have been satisfied, the algorithm fixes the sequence of the scheduled turbines to ensure that the turbines with the worst condition get maintenance first. This procedure is repeated every day until $t = T$. This is expressed in Algorithm 5.

Algorithm 5 Pseudo code for the implementation of algorithm for *Move turbine suggestion.*

```
if Number of turbines scheduled for d < n then
    Update the schedule by satisfying the d and n requirements;
for Every day where the daily capacity is exceeded do
    while There are more than four scheduled turbines on the given day do
        Get a moving suggestion for a turbine and a day from Algorithm 4;
        Move suggested turbine from current day to suggested day;
Use this new schedule, s_p, as placeholders;
Sort turbines from worst to best condition on the given day, t;
while Previous day do
    Add maintained turbines from s_p to updated schedule, s_u;
for Each sorted turbine that has not gotten maintenance do
    Place turbine into s_u based on placeholders from s_p;
return s_u;
```

8.2.3 Results and discussion

The main parameters that are used in this case study can be found Table D.1 (in Appendix). There are 32 turbines in the initial planning period, and they only need maintenance once in this maintenance period. This model only considers the logistic delay, and not the mean (active) repair time (from section 2.4), i.e., it assumes that a turbine can be fixed in a day and not the actual repair time. However, it is important to consider all aspects of the problem when optimizing. If purchasing was to be optimized and get the cheapest possible turbines, then perhaps the actual repair time will be longer, which will result in a higher cost. This could have been a further development of the proposed model to make it more refined. Furthermore, it is assumed that all planned maintenance days from the optimization model are available. The approach should be plausible since the activities are put out in time and scheduled for a day. Furthermore, there are restrictions such as d and n and the number of turbines that can be maintained per day.

The model has been run 10 times and calculated the cost of the final schedule for all of four different heuristics and a naive approach. This could be simulated several times to find the best heuristic. Furthermore, the initial conditions Y_0 and the degradation trajectories generated for the individual turbines vary for each test run. However, this model could have been tested with fixed initial conditions if they were known because this would reduce some of the uncertainty. Then the biggest uncertainty would be the rate of the degradation trajectories and the weather windows. It should be emphasized that for a specific situation, when the distribution over the Y_{i,t_0}-values in known, a company like Equinor (Equinor Renewables, 2022) could run simulations (as done in this report) to find the best heuristic for this initial condition. It should be noted that for other initial conditions, another heuristic may be best. The Python code from Appendixes B.3 and B.2 can be used to test out other heuristics with different d and n values.

Starting at the first day of the maintenance window, the *Naive approach* fills the day up with the four turbines with the worst conditions. This procedure is repeated until there are no turbines left that require maintenance, which is after eight days. The Python code for this naive approach is found at the end of Appendix B.3. The results are presented in Table 8.3. The best/ cheapest results of each test run are colored green, and the worst/ most expensive results are colored red in Table 8.3. All the results are rounded to the nearest 2 decimals. The table shows that *Heuristic 2* performed best on every single test run, and that all the heuristics performed better than the naive approach on every test run. For every test run, the different cost between the best and worst result was more than 10 on average. This shows that it can be clever to use a predictive maintenance model. *Heuristic 2* performs best under the given circumstances out of the four proposed heuristics. However, *Heuristic 2* is not guaranteed to propose the most optimal schedule globally. The average of the 10 rounds show that *Heuristic 3* performs worst of the four heuristics, followed by *Heuristic 4*, and then *Heuristic 1*. With the given criteria, number of turbines, and distribution of initial conditions Y_0, *Heuristic 3* performs the worst of the four heuristics. However, for some companies this heuristic might be the most robust one, because it has the most conservative criteria out of the four heuristics, meaning that it is taking the least risks. It is the one that forces the highest number of turbines to be maintained earlier. These models do not account for all uncertainties connected to performing maintenance at offshore wind farms. For instance, the weather could change which might force turbines to be delayed, there can be technical problems with the vessels. Furthermore, the degradation rate could go faster than expected on some turbines which might lead to an exceeded capacity some days. It is smart to do maintenance a little earlier on some turbines so that the risk of having to maintain too many turbines on day is reduced. This model shows that it can be cost-beneficial to perform CBM under selected heuristics.

Table 8.3: Comparison of cost results from four different heuristics and a naive approach

	Heuristic 1 (8 days, 16 turbines)	*Heuristic 2 (7 days, 10 turbines)*	*Heuristic 3 (7 days, 16 turbines)*	*Heuristic 4 (7 days, 14 turbines)*	*Naive approach*	*Difference between best and worst result*
Test 1	337.022	336.53	337.74	337.39	347.37	10.84
Test 2	333.92	333.92	333.97	333.92	343.66	9.74
Test 3	338.15	338.13	338.93	338.28	347.48	9.35
Test 4	338.34	337.28	339.37	338.54	350.17	12.89
Test 5	338.02	337.85	338.05	337.85	346.48	8.63
Test 6	336.15	336.15	336.15	336.15	342.25	6.1
Test 7	334.93	334.93	335.29	335.12	346.03	11.1
Test 8	337.40	336.50	338.56	337.78	348.84	12.34
Test 9	335.49	333.53	335.59	335.40	347.15	13.62
Test 10	338.02	337.37	338.22	337.64	348.85	11.48
Average	**336.744**	**336.219**	**337.187**	**336.807**	**346.828**	**10.609**

Chapter 9

Conclusions, Discussion, and Recommendations for Further Work

This chapter presents a discussion of the objectives, findings, and overall approach, followed by a summery and conclusion. Lastly, recommendations for further work are provided.

9.1 Discussion

Since there have been multiple discussions throughout this project thesis, the first part of this section will summarize the main points of the discussions. From the discussion of the literature review in section 5.2, all reviewed articles follow all the suggested steps from *Problem formulation - modeling* from section 4.1. In section 6.5.1, the model proposed in this report is compared to the models proposed in the literature review. The optimization models in the literature are much more complex than the proposed model in this report because the academic background of the reviewed authors is within optimization. The discussion in section 6.5.2 reflects over limitations to the proposed model. It is important to find good routes between the turbines. However, no distances are considered in the proposed model due to the complexity that it would entail. Furthermore, the problem is not as simple as finding the shortest distance between all turbines in a farm. Moreover, it is important to consider uncertainties such as weather which is done by the proposed model. This report's model returns one maintenance window per sub-model. It would have been more beneficial if it returned multiple weather windows so that the maintenance schedule was more flexible and could find more ideal times based on the condition of each turbine. Section 8.1.5 discusses the case study of *Approach 2*. One of the main challenges that the proposed model faces is that there is a big chance that the capacity is exceeded some days. A main reason for this is that the model schedules all turbines for the last day if they have not been maintained so far. Furthermore, the planning horizon can be challenging in stochastic programming because the degrading condition of a turbine might develop differently than anticipated. The proposed model focuses on finding the most cost-beneficial maintenance limit for individual turbines instead of having a wind farm perspective. The discussion goes on to address several challenges that were faced when implementing the proposed model which leads to the heuristics proposed in section 8.2. In section 8.2.3, a discussion is carried out about the proposed heuristics for *Approach 2*. All four heuristics perform better than the naive approach which is quite conservative. Even though *Heuristic 2* performs best on all the tests, it is not certain that this is the best of the four heuristics because these heuristics only handle some but not all possible uncertainties. *Heuristic 2* left the most flexibility at the end out of the four heuristics. However, in a real situation and with more uncertainties considered, it might be more beneficial to go with a more conservative heuristic such as *Heuristic 3*.

These next parts discuss the objectives of this project report. Here is a recap of the objectives, *OX*, from section 1.2: *O1*: Present the general framework of wind turbines, offshore wind farms, typical wind turbine failures and critical components. *O2*: Present the key concepts of PdM, PHM for RUL-estimation, and a classification of prognosis models. *O3*: Present the key concepts of optimization and modeling. *O4*: Perform an in-depth literature review on optimization models for routing and scheduling for offshore wind farms. *O5*: Create a simple optimization model for routing and scheduling at offshore wind farms under logistics constraints and real data given by Equinor (Equinor Renewables, 2022). *O6*: Create a catalog of ideas on how to merge optimization models with PdM models. *O7*: Implement some of the ideas from the catalog through case studies.

The first objective, *O1*, is achieved in Chapter 2, which introduces the industrial background of offshore wind farms. Furthermore, different types of faults in wind turbines such as generator faults, and the possible causes of the faults are presented in Table 2.1. It is shown that the duration of the downtime after a failure can vary depending on multiple factors such as the availability and capacity of the assets. This is a good objective because it gives the reader a starting point before they get into the more advanced parts of the thesis.

The second objective, *O2*, of this report is achieved in Chapter 3. PdM uses CBM to predict when an item will fail and is based on a prognosis for the degradation of the item. PHM has gained attention within the field of maintenance because it can be used for creating better maintenance schedules. PHM often includes data acquisition, diagnostics, prognostics, and health management. Diagnostics involves identifying and quantifying damages that have occurred, while prognostics involves trying to predict damages that have not yet occurred.

Finally, prognosis models can be classified into physics-based models and data-driven models. Data-driven models can again be divided into artificial intelligence approaches and statistical data-driven approaches. This distinction was quite important for the specialization project (Lien, 2021), but is not so much in focus in this project thesis. However, it is still important that the reader gets this theoretical background going into Chapter 7 and Chapter 8. The reason for this is that the Wiener process, which is used as the degradation model in the PdM model in the case study, is a statistical data-driven approach. Therefore, this objective is good.

The third objective, *O3*, is achieved in Chapter 4. The field of optimization is huge, and all of it cannot be covered in this chapter. However, most important concepts of optimization and modeling that were used in this project thesis are covered in this chapter. The main objective of this master's thesis is to suggest methods for combining optimization models with PdM models for offshore wind farm maintenance. Therefore, both *O2* and *O3* are good objectives. They give the reader a theoretical introduction to the main fields that are considered in this project.

The fourth objective, *O4*, is achieved in Chapter 5. Seven articles about optimization models for routing and scheduling for offshore wind farms have been studied. The "modeling recipe" from section 4.1 was used to get a deeper understanding of how optimization models could be constructed. Furthermore, the objective, the constraints, and the outputs of the proposed models were especially of interest when studying the models since the structure of performing these steps is quite consistent in the studied literature. It could be argued that the literature review was not in-depth since it is not very long. However, it was sufficient for its purpose which was to get a deeper understanding of how optimization models for routing and scheduling for offshore wind farms could be constructed. Therefore, *O4* is considered to be a good objective that fully achieved its purpose.

The fifth objective, *O5*, is achieved in Chapter 6. After becoming familiar with relevant theory and literature from achieving *O3* and *O4*, a simple optimization model could be created. The data provided by Equinor (Equinor Renewables, 2022) is simplified because it was important that the model did not get too advanced. The model is not the main contribution of this report and it would be way too time consuming. The weather data concerning wave height and wind speed was treated and used to create weather windows for the model. However, some data was treated but not used in the final model. For instance, the distance data and the attempts to find the shortest path were not included since it was unclear what kinds of shortest paths should be found. The problem at stake is not to be solved as a traveling salesman problem since all of the turbines at Dogger Bank A are most likely not going to be visited on the same trip. Furthermore, it would be beneficial to know how the optimization model could be connected to the PdM model beforehand so that it could be adjusted accordingly. However, the field of optimization had to be studied first to get an understanding of how to connect the two fields. The simple optimization model proposed in this report can be found in section 6.3. This objective is a good objective because it gives practical insight into the field of optimization and into challenges concerning taking logistics constraints and real data into account.

The sixth objective, *O6*, is achieved in Chapter 7.2. Some different approaches have been suggested on how optimization models could be merged with PdM models. Ideally, all the approaches would be tested through case studies. However, there was only enough time to create a case study of one of the approaches. Furthermore, each approach is described with a *problem statement*, *decision variables*, and *modeling challenges*. This way someone could further develop the original ideas, and perhaps be inspired to come up with new ideas on how to merge optimization models with PdM models. This objective was good and successful because it resulted in multiple different ideas which can be further developed or used for inspiration.

The seventh and final objective, *O7*, is attempted in Chapter 8. From section 7.2, a case study on *Approach 2* was implemented. Some attempts were made before arriving at the approach presented in section 8.1, which was quite time consuming. The resulting schedule that came from the model could not provide feasible solutions since the daily capacity was exceeded in the test runs. Furthermore, four different heuristics for *Approach 2* are tested in section 8.2 to get feasible solutions. The heuristic that performs the best under the given circumstances might be taking too much of a risk for a real case scenario. Therefore, it would be beneficial for a company to test out different heuristics such as the proposed ones under more realistic circumstances from their own data. All of this took more time than expected leaving no more time for implementing other approaches. Overall, this objective was good in that one of the approaches was tested out, but it was too ambitious.

To conclude the objectives, the overall approach was good because there was enough time to get an introduction into both the field of PHM from the specialization project (Lien, 2021) and the field of optimization. This was necessary to suggest alternative approaches on how the models could be tied together. However, it is difficult to say how good the suggested approaches are for tying optimization models with PdM models since most of them could not be tested. Furthermore, there was little pre-knowledge in the two fields before the specialization project and the master's thesis were started, nor were there any literature dealing with the same type of problem. This made it difficult to propose very credible suggestions. However, one of the approaches was tested in the case study which resulted in some promising results in Table D.4. These simulated results show that scheduling based on the condition of turbines is cheaper than being too conservative and scheduling every turbine as soon as possible at the beginning of the maintenance period. As discussed in section 8.2.3, even though *Heuristic 2* gave the best results in these tests, it might not be the best heuristic to use for Equinor (Equinor Renewables, 2022). The proposed heuristics still have problems such as not taking the traveling distance between two turbines into account, but this was not the main objective to be achieved. The main objective of this master's thesis is to suggest methods for combining optimization models for routing and scheduling, with PdM models for offshore wind farm maintenance. There is still more research and testing which can be done for this objective.

9.2 Summary and Conclusions

The main objective of this master's thesis is to suggest methods for combining optimization models for routing and scheduling, with PdM models for offshore wind farm maintenance. To reach the main objective, it is important to have knowledge within prognostics modeling and optimization modeling. From the specialization project (Lien, 2021), knowledge and practical experience were gained within the field of prognostics and degradation modeling through reviewing literature and performing a case study. To gain knowledge within the field of optimization that specifically concerns routing and scheduling, seven articles have been studied. When studying the articles, the main objective, the constraints, and the outputs of the presented optimization models were in focus. Furthermore, some experience has been gained within this field by treating data provided by Equinor (Equinor Renewables, 2022) in various ways such as finding daily weather windows that comply with given constraints, and finding the shortest path between all of the turbines. Then, a simple optimization model is created in section 6.3 which only considers one of the three wind farms at Dogger Bank. The objective of this model is to minimize the number of days offshore at Dogger Bank A under certain constraints. The run time of the entire model took too long to get optimal results, so the problem, i.e., the year, and the number of turbines, is divided into three and sub-optimized. The model returned one 14-day period per sub model. After knowledge and experience were gained within both fields, the main objective could be tackled.

In section 7.2, a catalog of four different approaches on combining optimization and PdM models is presented. *Approach 1* focuses on the planning perspective. It considers how to handle a situation where a turbine which is not scheduled for maintenance on this trip, has reached the maintenance limit and needs maintenance. *Approach 2* focuses more on condition-based maintenance. It considers how the turbines should be prioritized for maintenance daily based on their conditions and expected RUL. *Approach 3* considers whether or not a maintenance window should be extended with some days to complete maintenance on the turbines that have a higher chance of failing before they have been scheduled, or not. Finally, *Approach 4* considers whether or not any extra maintenance teams or DCs should be rented to finish maintenance earlier.

Approach 2 is put to the test in Chapter 8. This case study uses the results from the proposed optimization model for routing and scheduling in Chapter 6 as the time window within which to perform maintenance. Furthermore, it uses the PdM model in equation 7.5 to make schedules for the turbines based on their condition and expected RUL. This is a stochastic programming problem where there are not only two stages, but also a rolling horizon perspective on the stochastic programming. However, the results from running the model for *Approach 2* in section 8.1 do not return feasible schedules seeing that the daily capacity is in most cases exceeded and too many turbines get postponed until the last day of the maintenance period. If too many turbines are scheduled in the last days of the maintenance period, there is a good chance that there is not time to maintain all of them due to stochastic parameters such as the weather. Section 8.2 proposes four different heuristics to reduce the risk of exceeding the daily capacity on any given day. They still allow for a relatively flexible condition-based schedule so that healthy turbines can be postponed, and impaired turbines can be maintained earlier. Further, these heuristics consider the wind turbines to be a part of a bigger network to reduce the total cost instead of considering the best solution for each individual turbine. The only varying factors between the four heuristics are the length of the period within which some turbines are forced to be maintained, and the number of turbines that must be maintained within this period. The results from running the models for the four heuristics show that the suggested approach is better than taking a conservative approach and scheduling all turbines at the first and best opportunity one gets.

To conclude, large savings could be achieved by further developing these models by reducing down times, and optimizing schedules and asset utilization. To get these results more competence must be built within combining these fields for an optimal model and stronger and more realistic tools which can incorporate stochastic parameters such real condition and weather must be created. To achieve this, systems which can collect and deliver data must be built. The proposed optimization model is simple, and far from realistic. However, more advanced models have been created in the reviewed literature, which could be investigated further, but these still need to be tied in a sensible way to prognostic models. The scheduling model created in this report could be used with realistic data, such as real conditions and real degradation rates, to test out different heuristics. Unlike in this report, the model would have to be run many times to ensure that the results are valid. However, since weather is not being considered the model would have to be improved before using it to create actual schedules. One possible approach to consider the weather is to plan more conservatively. Furthermore, all of the requirements from Equinor (Equinor Renewables, 2022) have not been met, and require more research. Overall, more work and competence are required withing the fields of optimization, predictive maintenance, and the combination of the two fields. Some recommendations for further work within these fields has been proposed in the following section.

9.3 Recommendations for further work

Possible extensions and recommendations for future work are presented in the following. The aim is to provide some guidance for further analysis and topics that need to be addressed for successfully combining PdM and optimization models for routing and scheduling.

Optimization. The optimization model proposed in section 6.3 only considers one wind farm and the SOV which is relatively simple. The ideal optimization model should consider multiple wind farms, multiple vessels (DCs), as many stochastic variables as possible, and ideal traveling routes. Furthermore, the model should be more flexible and return multiple optional maintenance windows that are either optimal or close to optimal so that the condition of the turbines can have a bigger influence on when maintenance is performed. For even more flexibility, it would be beneficial if the model could consider extending the length of the maintenance window if necessary and possible. Overall, the problem with the optimization models is that they use overly simplified environmental, load and degradation models. Furthermore, they can rarely optimize from a great number of objectives concurrently.

Predictive maintenance. The wind industry still does not have optimal PdM models. In this project it proved difficult to have a wind farm perspective instead of just considering individual turbines, and it was difficult to consider multiple wind farms. In the bigger picture, the industry is missing a unified real-time framework for integrating RUL, weather windows, logistic resources, and production profiles. This would require hybrid uncertainty models that are based on the physics of failures and on data-driven approaches.

Combining the two models. To combine an optimization model for routing and scheduling with a PdM model, one must have a good understanding of both fields. In the future, these two fields should be studied more in depth simultaneously to gain an in-depth understanding of them. Approaches *1*, *3*, and *4* from the catalog in section 7.2 could be explored and tested to either find good solution methods or to obtain inspiration for new ideas on combining the models. In the future, the ideal goal is to have digital twins with data and real-time models that can model unknowns in the context of wind energy. The optimization model could take care of the logistics such as the actual route, the weather conditions, and the cost of utilizing assets. Moreover, the PdM model could keep track of the aspects related to PHM such as the actual component state, the physical load on the turbines, and the actual maintenance carried out. These models could interact via the Internet of Things and make decisions based on the data from the other model. More research within the above-mentioned fields and artificial intelligence will be required to achieve this.

Appendix A

Acronyms

CBM Condition-based maintenance

CM Corrective maintenance

DC Daughter craft

O&M Operations and maintenance

PdM Predictive maintenance

PHM Prognosis and health management

PM Preventive maintenance

RUL Remaining useful life

SOV Service operation vessel

Appendix B

Python code

B.1 Python code for optimization model in Chapter 6

```
from re import M
import gurobipy as gp
from gurobipy import GRB

# Initiate problem
m = gp.Model("TurbineOptimization")

# PARAMETERS

days = []
for day in range(0,365): days.append(day)

turbines = [] # Turbines in prioritized order
for turbine in range(0,96): turbines.append(turbine)
turbines = turbines

number_of_maintenance_teams = 4
required_maintenance_hours_per_turbine_per_year = 20
max_workload_per_day = 13
big_m = 100
scenarios = [0,1,2]
```

```
# List of available working hours with wave height, wind speed and 2
    states considered. The lists are shortened from having 365 elements.

weather_window_1990 = [18, 22, 22, 13, 23, ..., 0, 0, 21]
weather_window_2003 = [22, 22, 22, 22, 22, ..., 21, 22, 20]
weather_window_2018 = [15, 22, 0, 22, 22, ..., 0, 22, 22]

weather_window = [weather_window_1990, weather_window_2003,
    weather_window_2018]

# 1990 = available_working_hours[0], 2003 = available_working_hours[1],
    2018 = available_working_hours[2]
available_working_hours = [[],[],[]]

for scenario in scenarios:
  for window in weather_window[scenario]:
    available_working_hours[scenario].append(window *
        number_of_maintenance_teams)

# DECISION VARIABLES

is_offshore = {}
daily_work_hours = {} # dictionaries
total_maintenance_hours = {}
ensure_sequence = {}
fourteen_days_offshore = {}
twentyeight_days_onshore = {}

# Saves all variables in a dictionary and adds them to the model
for day in days:
  is_offshore[day] = m.addVar(vtype = GRB.BINARY, name="NA")

for day in days:
  for turbine in turbines:
    for scenario in scenarios:
      daily_work_hours[(day,turbine,scenario)] = m.addVar(vtype = GRB.
```

```
            CONTINUOUS, name="NA")

for day in days:
  for turbine in turbines:
    for scenario in scenarios:
      total_maintenance_hours[(day,turbine,scenario)] = m.addVar(vtype =
          GRB.CONTINUOUS, name="NA")

for day in days:
  for turbine in turbines[:-4]:
    ensure_sequence[(day,turbine)] = m.addVar(vtype = GRB.BINARY, name="NA
        ")

for day in days[1:-13]:
  fourteen_days_offshore[(day)] = m.addVar(vtype = GRB.BINARY, name="NA")

for day in days[1:-41]:
  twentyeight_days_onshore[(day)] = m.addVar(vtype = GRB.BINARY, name="NA"
      )

# CONSTRAINTS

# first day cannot be offshore
i = days[0]
m.addConstr(is_offshore[i] == 0, "NA")

# Max workload for teams
for day in days:
  for scenario in scenarios:
    m.addConstr(gp.quicksum(daily_work_hours[(day, turbine, scenario)] for
        turbine in turbines) <= max_workload_per_day *
        number_of_maintenance_teams, "NA")

# Max 13 hours on turbine in a day
for turbine in turbines:
  for day in days:
    for scenario in scenarios:
```

```
            m.addConstr(daily_work_hours[(day, turbine, scenario)] <=
                max_workload_per_day, "NA")

# Cannot use more than available hours in a day
for day in days:
    for scenario in scenarios:
        m.addConstr(gp.quicksum(daily_work_hours[(day, turbine, scenario)] for
            turbine in turbines) <= available_working_hours[scenario][day] *
            is_offshore[day], "NA")

# Fixing is done in specific order
for day in days:
    for turbine in turbines[:-1]:
        for scenario in scenarios:
            m.addConstr(total_maintenance_hours[(day, turbine, scenario)] >=
                total_maintenance_hours[(day, turbine + 1, scenario)], "NA")

end_of_period = days[-1]

# Required maintenance hours in a year
for turbine in turbines:
    for scenario in scenarios:
        m.addConstr(total_maintenance_hours[(end_of_period, turbine, scenario)
            ] == required_maintenance_hours_per_turbine_per_year, "NA")

# Find total maintenance for every day throughout year
for turbine in turbines:
    for day in days:
        for scenario in scenarios:
            m.addConstr(total_maintenance_hours[(day, turbine, scenario)] == gp.
                quicksum(daily_work_hours[(iter_day, turbine, scenario)] for
                iter_day in days[:day+1]), "NA")

# Ensure that task w is finished before task w+4 is started
for day in days:
    for turbine in turbines[:-4]:
        for scenario in scenarios:
```

```
        m.addConstr(required_maintenance_hours_per_turbine_per_year -
            total_maintenance_hours[(day, turbine, scenario)] <= big_m *
            ensure_sequence[day, turbine], "NA")
        m.addConstr(daily_work_hours[(day,turbine + 4, scenario)] <= big_m *
            (1 - ensure_sequence[(day,turbine)]), "NA")

# 14 days offshore in a row
for day in days[1:-13]:
    m.addConstr(13 - gp.quicksum(is_offshore[(iter_day)] for iter_day in
        days[day+1:day+14]) <= big_m * fourteen_days_offshore[day], "NA")
    m.addConstr(is_offshore[day] - is_offshore[day - 1] <= big_m * (1 -
        fourteen_days_offshore[day]), "NA")

# 28 days onshore after 14 days offshore
for day in days[1:-41]:
    m.addConstr(gp.quicksum(is_offshore[(iter_day)] for iter_day in days[
        day+14:day+42]) <= big_m * twentyeight_days_onshore[day], "NA")
    m.addConstr(is_offshore[day] - is_offshore[day - 1] <= big_m * (1 -
        twentyeight_days_onshore[day]), "NA")

# OBJECTIVE FUNCTION

number_of_offshore_days = gp.quicksum(is_offshore[(day)] for day in days)
m.setObjective(number_of_offshore_days, GRB.MINIMIZE)
m.optimize()

print("Objective value")
print(m.getObjective().getValue())

# PRINT OFFSHORE DECISION VARIABLES

for day in days:
    for turbine in turbines:
        if daily_work_hours[(day, turbine, scenario)].x > 0:
            print("Day: ", day + 1, " , Turbine: ", turbine + 1, " , Hours
                : ", daily_work_hours[(day, turbine, scenario)].x)
```

B.2 Python code for Approach 2 in Section 8.1

```
from pickle import FALSE, TRUE
import matplotlib.pyplot as plt
import numpy as np
import math

t_start = 104
t_final = 118
N = 1008
dt = (t_final-t_start)/N
n = 32
x = 88
MTTF = 100
l = 100
c_R = 1000
c_F = 10000
c_U = 1000
mu = 1
sigma = 1

dW, W, Y = [], [], [] # n-by-N, n-by-N, n-by-N+1 matrix
for i in range(n):
  list_in_matrix_dW, list_in_matrix_W, list_in_matrix_Y = [], [], []
  for j in range(N):
    list_in_matrix_dW.append(0)
    list_in_matrix_W.append(0)
  dW.append(list_in_matrix_dW)
  W.append(list_in_matrix_W)
  for j in range(N+1):
    list_in_matrix_Y.append(0)
  Y.append(list_in_matrix_Y)

t = [] # time vector
for i in range(N+1): t.append(t_start + i*dt)

# Generate the degradation paths for n turbines for N stpes
```

```
for unit in range(n):
  dW[0][0] = math.sqrt(dt) * np.random.randn() # first approximation
      outside loop
  W[0][0] = dW[0][0]
   for step in range(1, N):
     dW[unit][step] = math.sqrt(dt) * np.random.randn() # generate
         increments
     W[unit][step] = W[unit][step-1] + dW[unit][step]
   for i in range(N+1):
       Y[unit][i] = x + mu*t[i] + sigma*0 - t_start if i == 0 else x + mu*t
           [i] + sigma*W[unit][i-1] - t_start

def calculate_alpha_ml(m): return (1-m)/mu
def calculate_beta_ml(m): return (1-m)**2/(sigma)**2

# F(t;alpha;beta)
def CDF_IG(x, alpha, beta):
   sqrt_beta = math.sqrt(beta)
   sqrt_x = math.sqrt(x)
   if x <= 0: return 0
   half_equation = phi(-sqrt_beta*sqrt_x/alpha-sqrt_beta/sqrt_x)
   if half_equation > 0:
     half_equation = math.log(half_equation)
     half_equation = math.exp(half_equation + 2 * beta/alpha)
   return phi(sqrt_beta*sqrt_x/alpha-sqrt_beta/sqrt_x) + half_equation

# Approximation to Standard Normal Probability
def phi(x):
   if abs(x) < 13:
     k = 1/(1+0.2316419*abs(x))
     c1 = 0.31938153
     c2 = -0.356563782
     c3 = 1.781477937
     c4 = -1.821255978
     c5 = 1.330274429
     POLYNM = k*(c1+k*(c2+k*(c3+k*(c4+k*c5))))
     f_x = math.exp(-x**2/2) * 1/math.sqrt(2*math.pi)
```

```
        phi_x = f_x * POLYNM
        if (x >= 0): phi_x = 1 - phi_x
    else:
        phi_x = 0 if x < 0 else 1
    return phi_x

def calculate_MTBR(m, T_L): return m/mu+T_L

# Finding the C(m) = cost equation
def cost_equation(m, T_L):
    alpha = calculate_alpha_ml(m)
    beta = calculate_beta_ml(m)
    return (c_R + c_F*CDF_IG(T_L, alpha, beta) + c_U*NumInt(0, T_L, alpha,
        beta, T_L)) / calculate_MTBR(m, T_L)

# Simple numerical integration routine, a=start, b=end
def NumInt(a, b, p1, p2, p3):
    nSteps = 100
    dx = (b - a) / nSteps
    x = 0
    Sm = 0
    for j in range(1, nSteps, 1):
        Sm = Sm + dx * 0.5 * (integrand(x, p1, p2, p3) + integrand(x + dx, p1,
            p2, p3))
        x = x + dx
    return Sm

def integrand(x, alpha, beta, T_L):
    return PDF_IG(x, alpha, beta) * (T_L - x)

# f(t;alpha;beta)
def PDF_IG(x, alpha, beta):
    return math.sqrt(beta/(2*math.pi*x**3)) * math.exp(-beta*(x-alpha)
        **2/(2*alpha**2*x)) if x > 0 else 0

def create_leadtimes_and_costs_dicts():
    leadtimes_dict, costs_dict = {}, {} # {m : [leadtimes]}, {m : [costs]}
```

```
  for m_l in range(x-5, l):
    leadtimes_dict[m_l], costs_dict[m_l] = [], []
    leadtime = 0
    while leadtime <= (l-1) - m_l:
      cost = cost_equation(m_l, leadtime)
      leadtimes = leadtimes_dict.get(m_l)
      leadtimes.append(leadtime)
      leadtimes_dict[m_l] = leadtimes
      costs = costs_dict.get(m_l)
      costs.append(cost)
      costs_dict[m_l] = costs
      leadtime += 1
  return [leadtimes_dict, costs_dict]

# {turbine : [condition]}, index indicates day, i.e. i=0 -> day=104
Y_dict = {}
for unit in range(n):
  condition_list = []
  for i in range(N+1):
    if t[i].is_integer():
      condition_list.append(Y[unit][i])
      Y_dict[unit+1] = condition_list

# Make a priority ordered list from {turbine : [condition]}
def make_priority_sequence(dict_with_conditions, day_to_sort_by):
  day_to_sort_by -= t_start
  return_dict = {} # {turbine : sorted_day_condition} -> from worst
     condition to best condition
  for turbine, conditions in dict_with_conditions.items():
    return_dict[turbine] = conditions[day_to_sort_by]
  return_dict = {k: v for k, v in sorted(return_dict.items(), key=lambda
     item: item[1], reverse=True)} # sort dict
  return return_dict

# Find approxomate condition by rounding the float-conditions
def create_condition_approxomation_dict(priority_dict):
  approximation_dict = {} # {turbine : condition}
```

```
  for turbine, condition in priority_dict.items():
    approximation_dict[turbine] = round(condition)
  return approximation_dict

# Finds most cost-beneficial lead time for different conditions
def find_best_leadtimes_and_costs():
  leadtimes_and_costs_dicts = create_leadtimes_and_costs_dicts()
  leadtimes_dict = leadtimes_and_costs_dicts[0]
  costs_dict = leadtimes_and_costs_dicts[1]
  return_dict = {} # {condition : [leadtime, cost]}
  for condition, costs in costs_dict.items():
    min_cost = min(costs)
    min_index = costs.index(min_cost)
    leadtimes = leadtimes_dict.get(condition)
    min_leadtime = leadtimes[min_index]
    return_dict[condition] = [min_leadtime, min_cost]
  return return_dict

# Creates initial maintenance schedule (from first day in maintenance
    period) for the different turbines based on their condition
def create_inital_schedule():
  day = t_start+1
  schedule = {} # {day : [turbines]}
  for i in range(day, t_final+1):
    schedule[i] = []
  best_leadtimes_and_costs = find_best_leadtimes_and_costs() # {condition
     : [leadtime, cost]}
  priority_sequence = make_priority_sequence(Y_dict, day)
  approxomation_dict = create_condition_approxomation_dict(
     priority_sequence) # {turbine : condition}
  for turbine, condition in approxomation_dict.items():
    leadtime_cost_list = best_leadtimes_and_costs.get(condition)
    leadtime = leadtime_cost_list[0]
    current_schedule = schedule.get(day+leadtime)
    current_schedule.append(turbine)
    schedule[day+leadtime] = current_schedule
  return schedule
```

```
# update the schedule every day for most ideal sequence
def update_schedule(schedule, current_day, maintained_turbines):
  if len(maintained_turbines) >= n:
    return [schedule, current_day, maintained_turbines]
  new_schedule = {} # {day : [turbines]}
  for i in range(current_day, t_final+1):
    new_schedule[i] = []
  best_leadtimes_and_costs = find_best_leadtimes_and_costs()
  p = make_priority_sequence(Y_dict, current_day)
  approxomation_dict = create_condition_approxomation_dict(p)
  maintained_today = False
  for turbine, condition in approxomation_dict.items():
    if turbine not in maintained_turbines:
      leadtime = best_leadtimes_and_costs.get(condition)[0]
      if leadtime == 0:
        maintained_today = True
        maintained_turbines.append(turbine)
      that_day = current_day+leadtime if (current_day+leadtime) <= t_final
          else t_final
      if (current_day == t_final):
        if turbine not in maintained_turbines: maintained_turbines.append(
            turbine)
        that_day = t_final
      schedule_new = new_schedule.copy().get(that_day)
      schedule_new.append(turbine)
      schedule[that_day] = schedule_new.copy()
  if not maintained_today:
    schedule[current_day] = []
  while current_day < 118 and len(maintained_turbines) <= n:
    current_day += 1
    update_schedule(schedule, current_day, maintained_turbines)
  return [schedule, current_day, maintained_turbines]

# Test:
maintained_turbines = []
inital_schedule = create_inital_schedule()
```

```
final_schedule = update_schedule(inital_schedule, 105, maintained_turbines
    )[0]
print("result:␣", final_schedule)

# Plots
for p in range(n):
  plt.plot(t, Y[p], label="T" + str(p+1))
  plt.legend(loc="upper␣center", bbox_to_anchor=(0.5, 1.15), ncol=12)
dict1 = {"l": [], "due_date_limit": []}
for i, list_ in dict1.items():
  for j in range(N+1):
    list_.append(100) if i == "l" else list_.append(96)
plt.plot(t, dict1.get("due_date_limit"), "b-") # best maintenance limit
   for 0 lead time
plt.plot(t, dict1.get("l"), "r-") # failure limit
plt.xlabel('Time,␣t')
plt.ylabel('Condition,␣Y_t')
plt.minorticks_on()
plt.grid(True, "both")
plt.xlim([105, 118])
plt.show()
```

B.3 Python code for Heuristics in Section 8.2

The start of the code with the same code as in Appendix B.2 until the *create_initial_schedule()* function from "### CODE FROM APPENDIX B.2". Additionally, instead of a fixed initial condition *x*, this approach assigns a distribution of initial conditions in the *initial_conditions* list. Furthermore, *t_start* is now on the first day of maintenance offshore. In the following is the Python code for the heuristics:

```
import random
t_start = 105
t_final = 118
N = 1001
initial_conditions = [random.randint(80, 96) for i in range(n)]
```

```
### CODE FROM APPENDIX D

# Creates initial maintenance schedule (from first day in maintenance
    period) for the different turbines based on their condition
def create_new_initial_schedule(today, old_schedule):
  new_schedule = dict((d, []) for d in range(t_start, t_final+1))
  best_leadtimes_and_costs = find_best_leadtimes_and_costs() # {condition
      : [leadtime, cost]}
  priority_sequence = make_priority_sequence(Y_dict, today)
  approxomation_dict = create_condition_approxomation_dict(
      priority_sequence) # {turbine : condition}
  for maintained_day in range(t_start, today):
    new_schedule[maintained_day] = old_schedule.get(maintained_day)
    for turbine in old_schedule.get(maintained_day):
      approxomation_dict.pop(turbine)
  for turbine, condition in approxomation_dict.items():
    leadtime = best_leadtimes_and_costs.get(condition)[0]
    new_schedule_day = today+leadtime if today+leadtime <= t_final else
        t_final
    current_schedule = new_schedule.get(new_schedule_day)
    current_schedule.append(turbine)
    new_schedule[new_schedule_day] = current_schedule
  return new_schedule

# Count the number of turbines in 1st period
def check_min_turbines_first_period(schedule, first_period):
  counter, turbines_in_first_period = 0, 0
  for turbines in schedule.values():
    if counter < first_period:
      turbines_in_first_period += len(turbines)
    counter += 1
  else:
    return turbines_in_first_period

# Satisfy the minimum requirement of the first period, return updated
    schedule
```

```
def satisfy_min_turbines_first_period(schedule, min_turbines_first_period,
    first_period):
  counter = 0
  num_turbines_to_move = min_turbines_first_period -
      check_min_turbines_first_period(schedule, first_period)
  for day in range(t_start + first_period, t_final + 1):
    if counter == num_turbines_to_move:
      break
    turbines = schedule.get(day)
    while turbines and not counter == num_turbines_to_move:
      move_turbine = turbines[0]
      last_day_first_period_schedule = schedule.get(t_start + first_period
          - 1)
      last_day_first_period_schedule.append(move_turbine)
      schedule[t_start + first_period - 1] =
          last_day_first_period_schedule
      turbines.remove(move_turbine)
      schedule[day] = turbines
      counter += 1
      if counter == num_turbines_to_move:
        break
  return schedule

def find_over_filled_days(schedule):
  over_filled_days = []
  for day, turbines in schedule.items():
    if len(turbines) > 4:
      over_filled_days.append(day)
  return over_filled_days

# Check what turbines are cheapest/possible to move, return [move_day,
    move_turbine], assumes that turbines are placed according to condition
def move_turbine_suggestion(today, maintenance_day, turbines, schedule,
    min_turbines_first_period, first_period):
  sickest_turbine, healthiest_turbine = turbines[0], turbines[-1]
  # SPECIAL CASES:
```

```
if maintenance_day == today: # first day in period: helthy turbine is
    scheduled later
  for day in range(maintenance_day + 1, t_final + 1):
    if len(schedule.get(day)) < 4:
      return [day, healthiest_turbine]
if maintenance_day == t_final: # last day: sickest turbine is scheduled
    earlier
  for day in range(maintenance_day - 1, today - 1, -1):
    if len(schedule.get(day)) < 4:
      return [day, sickest_turbine]
if maintenance_day == t_start + first_period - 1 and
    check_min_turbines_first_period(schedule, first_period) <=
    min_turbines_first_period: # last day in first period && first period
     <= 16 turbines: sickest turbine is scheduled earlier
  for day in range(t_start + first_period - 2, today - 1, -1):
    if len(schedule.get(day)) < 4:
      return [day, sickest_turbine]
# Move sick turbine earlier or healthy turbine later?
leadtime = maintenance_day - today
sick_condition = Y_dict[sickest_turbine][leadtime]
sick_cost = cost_equation(sick_condition, leadtime)
new_sick_cost = 100 # if so, there are no available days before this day
    ...
move_day_sick = maintenance_day
for day in range(maintenance_day - 1, today - 1, -1):
  if len(schedule.get(day)) < 4:
     new_sick_cost = cost_equation(sick_condition, day - today)
     move_day_sick = day
     break
healthy_condition = Y_dict[healthiest_turbine][leadtime]
healthy_cost = cost_equation(healthy_condition, leadtime)
new_healthy_cost = 100
move_day_healthy = maintenance_day
if check_min_turbines_first_period(schedule, first_period) >
    min_turbines_first_period: # must be at least x turbines in first
    period to move turbines later
  for day in range(maintenance_day + 1, t_final + 1):
```

```
            if len(schedule.get(day)) < 4:
                new_healthy_cost = cost_equation(healthy_condition, day - today)
                move_day_healthy = day
                break
    delta_sickest_turbine = new_sick_cost - sick_cost
    delta_healthiest_turbine = new_healthy_cost - healthy_cost
    return [move_day_sick, sickest_turbine] if delta_sickest_turbine <
        delta_healthiest_turbine else [move_day_healthy, healthiest_turbine]

# move turbine as the function suggests in a new function
def move_turbines(today, schedule, min_turbines_first_period, first_period
    ):
    if check_min_turbines_first_period(schedule, first_period) <
        min_turbines_first_period:
        schedule = satisfy_min_turbines_first_period(schedule,
            min_turbines_first_period, first_period)
    over_filled_days = find_over_filled_days(schedule)
    for over_full_day in over_filled_days:
        while len(schedule.get(over_full_day)) > 4:
            turbines = schedule.get(over_full_day)
            move_suggestion = move_turbine_suggestion(today, over_full_day,
                turbines, schedule, min_turbines_first_period, first_period)
            move_day, move_turbine = move_suggestion[0], move_suggestion[1]
            # Move turbine, move_turbine, to new day, move_day
            move_day_schedule = schedule.get(move_day)
            move_day_schedule.append(move_turbine)
            schedule[move_day] = move_day_schedule
            # Remove turbine, move_turbine, from old day, over_full_day
            over_full_schedule = schedule.get(over_full_day)
            over_full_schedule.remove(move_turbine)
            schedule[over_full_day] = over_full_schedule
    return schedule

def fix_sequence(today, schedule, min_turbines_first_period, first_period)
    :
    updated_placeholders = move_turbines(today, schedule,
        min_turbines_first_period, first_period)
```

```
    updated_schedule = dict((i, []) for i in range(t_start, t_final+1))
    priority_sequence = make_priority_sequence(Y_dict, today) # {turbine :
        condition}
    # remove turbines that have gotten maintenance ?
    day = t_start
    while day < today: # aka. previous days
        updated_schedule[day] = schedule.get(day)
        for turbine in schedule.get(day):
            priority_sequence.pop(turbine)
        day += 1
    for turbine in priority_sequence.keys():
        day += 0 if len(updated_schedule.get(day)) < len(updated_placeholders.
            get(day)) else 1
        while not updated_placeholders.get(day):
            day += 1
        scheduled_turbines = updated_schedule.get(day)
        scheduled_turbines.append(turbine)
        updated_schedule[day] = scheduled_turbines
    return updated_schedule

#Returns schedule and cost of heuristic
def run_heuristic_number(heuristic_number):
    heuristic_dict = {1: [16, 8], 2: [10, 7], 3: [16, 7], 4: [14, 7]}
    min_turbines_first_period = heuristic_dict.get(heuristic_number)[0]
    first_period = heuristic_dict.get(heuristic_number)[1]
    prev_schedule = create_new_initial_schedule(t_start, {})
    prev_fixed = fix_sequence(105, prev_schedule, min_turbines_first_period,
        first_period)
    for day in range(t_start+1, t_final+1):
        schedule = create_new_initial_schedule(day, prev_fixed)
        fixed_schedule = fix_sequence(day, schedule, min_turbines_first_period
            , first_period)
        prev_schedule, prev_fixed = schedule, fixed_schedule
    total_cost = 0
    for day, turbines in fixed_schedule.items():
        for turbine in turbines:
            conditions = Y_dict.get(turbine)
```

```
      condition = conditions[day-t_start]
      temp_cost = cost_equation(condition, 0)
      total_cost += temp_cost
  return [fixed_schedule, total_cost]

for i in range(4):
  h = run_heuristic_number(i+1)
  print("Heuristic", i+1, ", schedule: ", h[0], ", cost: ", h[1])


### Naive approach: fill first days with turbines based on the ones with worst condition
naive_schedule = dict((d, []) for d in range(t_start+1, t_final+1))
naive_maintained = []

for day in range(t_start+1, t_final+1):
  pri_seq_dict = make_priority_sequence(Y_dict, day) # returns {turbine : condition}
  listed = []
  for turbine, condition in pri_seq_dict.items():
    listed = naive_schedule.get(day)
    if len(listed) < 4 and turbine not in naive_maintained:
      listed.append(turbine)
      naive_schedule[day] = listed
      naive_maintained.append(turbine)

# count number of turbines in schedule + cost of this schedule:
total_cost = 0
for day, turbines in naive_schedule.items():
  for turbine in turbines:
    conditions = Y_dict.get(turbine)
    condition = conditions[day-t_start]
    temp_cost = cost_equation(condition, 0)
    total_cost += temp_cost

print("Naive schedule: ", naive_schedule, ", cost: ", total_cost)
```

Appendix C

Predictive maintenance model from specialization project

This section gives a short summary of the PdM model based on the stochastic Wiener process used in the case study of Lien (Lien, 2021). The model is inspired by Vatn (Vatn, 2020a).

Predictive maintenance model

The following situation is considered. (1) It is assumed that the degradation process can be observed continuously without any uncertainty. (2) A failure will occur if $X_t \geq l$ for some time t. (3) A request to replace the component with a new component will be placed when degradation approaches the failure limit. (4) The lead time, T_L is assumed to be stochastic, and can either be short, medium or long. (5) Finally, the objective is to determine the maintenance limit $m < l$, meaning how close the maintenance limit should be to the failure limit. The main different between the model proposed by Lien (Lien, 2021) and Vatn (Vatn, 2020a) is that the first model considers stochastic lead times, while the second model considers deterministic lead times. The cost equation that is being minimized is:

$$C(m) = \frac{c_\mathrm{R} + c_\mathrm{F} F(T_\mathrm{L}|m) + c_\mathrm{U} \int_0^{T_\mathrm{L}} f(t|m)(T_\mathrm{L} - t)\,dt}{\mathrm{MTBR}(m)}, \tag{C.1}$$

where c_R is the cost of renewal or replacement. c_F denotes the cost of failure. This parameter considers additional costs for CM and extra costs for the failure event. After this repair, the state of the item is assumed to be "good as new". c_U is the cost per hour of down time. $F()$ and $f()$

are respectively the CDF and the PDF for RUL, given that we are at the maintenance limit m at some point. Finally, MTBR(m) is the Mean Time Between Renewals, given the decision rule to request a maintenance at m.

The stochastic lead time is found by calculating the expected value of three different lead times: a short, a medium, and a long lead time. Each of these three deterministic lead times have a corresponding probability p_i, which indicates the likelihood of having that lead time. The expected costs for each maintenance limit are calculated by taking the weighted sum of the costs for short, medium, and long lead times multiplied with the corresponding probability:

$$C(m) = \mathrm{E}[C(m, T_{\mathrm{L}})] = \sum_{i=1}^{3} p_i C(m, t_{\mathrm{L},i}), \tag{C.2}$$

where 1 = short t_{L}, 2 = medium t_{L}, and 3 = long t_{L}.

In the PdM-model in equation C.1, one maintenance cycle is considered. $Y_t = m$ is assumed to be observed at time t in this cycle, while RUL_m is the time from t until a failure occurs. RUL_m is inverse-Gauss distributed with parameters $\alpha_m = (l-m)/\mu$ and $\beta_m = (l-m)^2/\sigma^2$, where μ and σ^2 are the parameters from the Wiener process, and l is the failure threshold. Furthermore, $F() = F(t;\alpha_m;\beta_m)$ is given by equation 7.3 and $f() = f(t;\alpha_m;\beta_m)$ is given by equation 7.2. The numerator of $C(M)$ is obtained by numerical integration. Finally, $\mathrm{MTBR}(m) = m/\mu + T_{\mathrm{L}}$.

Results

The results obtained in the case study of Lien, show that lead times affect the costs of maintenance and when maintenance should be performed, see Table C.1. It shows that maintenance costs are high, and the maintenance limits are shortest when the lead time is long, while when the lead time is short, costs are lowest and the maintenance limit is the highest. In addition, it shows that the expected costs are highest when the lead times are uncertain.

Table C.1: Optimal maintenance limit for short, medium, long and stochastic lead times

Lead times	$t_{\mathrm{L},1} = 2.5$	$t_{\mathrm{L},2} = 5$	$t_{\mathrm{L},3} = 7.5$	T_{L}
Optimal maintenance limit	92.1	87.6	83.6	84.7
Corresponding cost	10.624	10.875	11.070	11.269

Appendix D

Extra material for case study in Chapter 8

D.1 Expected costs and parameter values

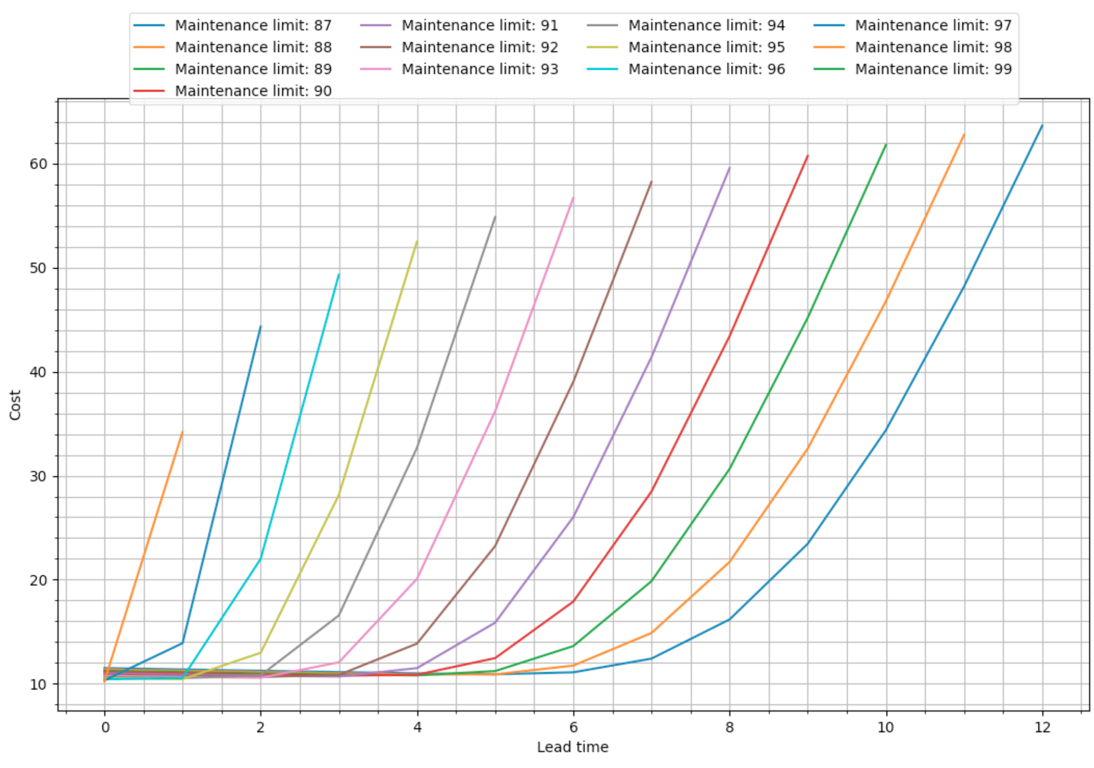


Figure D.1: Expected costs of lead times with varying maintenance limits

Table D.1: The parameter values used as a starting point for the case study.

Symbol in code	*Meaning of symbol*	*Value used in case study*
t_start, t_final	Time interval	104, 118
N	Number of steps to compute	1008
dt	Time step	14/1008
n	Number of turbines	32
x	Starting condition, Y_0	88
MTTF	Mean time to failure	100
l	Failure limit	100
c_R	Cost of replacement	1000
c_F	Cost of failure	10000
c_U	Cost of down time	1000
mu	μ	1
sigma	σ	1

D.2 Step-by-step example

For this step-by-step example of running the model, the number of turbines is reduced from 32 to 6, and the final day is $t = 108$ instead of $t = 118$ for the sake of simplicity. Assume that turbines number 1, 2, ..., 6 are going to be scheduled for maintenance during the days $t = 105, 106, ..., 108$ by using this model.

Table D.2: Most cost-beneficial lead times according to cost equation 7.5

Conditions, $Y_{i,t}$	Most cost-beneficial lead time, $x_{i,t}$
92, 93	2
94, 95	1
96, >96	0

Table D.3 shows that all turbines are assumed to have the same initial condition. On day $t = 105$ the condition of each individual turbine is checked, and the turbines are sorted accordingly: $[2,3,4,1,5,6]$. Then the condition of each individual turbine is rounded to the nearest integer and the most cost-beneficial lead times of each condition is found, see Table D.2. The turbines are placed accordingly:

```
schedule on day 105: {105: [2], 106: [3], 107: [4, 1, 5, 6], 108: []}
```

Table D.3: Daily conditions for example D.2

Day \Condition	$Y_{1,t}$	$Y_{2,t}$	$Y_{3,t}$	$Y_{4,t}$	$Y_{5,t}$	$Y_{6,t}$
$t_0 = 104$	91	91	91	91	91	91
$t = 105$	92.2	95.6	93.5	92.3	92.1	92.0
$t = 106$	94.1	-	95.1	94.3	93.8	92.3
$t = 107$	93.9	-	96.5	97.4	93.6	96.4
$t = 108$	96.5	-	-	-	93.4	-
Final condition	96.5	95.6	96.5	97.4	93.4	96.4

Since turbine 2 is scheduled for maintenance on day $t = 105$ today, it will receive maintenance today and it will not be rescheduled further in this period. Moving on, the current day is $t = 106$. The condition of each individual turbine is checked and rounded, and the turbines are placed accordingly (see Tables D.2, D.3):

```
schedule on day 106:  {105: [2], 106: [], 107: [3, 4, 1, 5], 108: [6]}
```

There are no turbines scheduled for maintenance on day $t = 106$ today. Turbine 3 was originally scheduled to maintenance today, but its condition allowed it to wait another day. Turbine 6 was originally scheduled for day $t = 107$, but the condition was good enough to postpone the maintenance with an extra day. The current day is now $t = 107$. The conditions of the turbines are checked and rounded, and the turbines are placed accordingly:

```
schedule on day 107:  {105: [2], 106: [], 107: [4, 3, 6], 108: [1, 5]}
```

Since turbines 3, 4, and 6 are scheduled for maintenance on day $t = 107$, they will receive maintenance today. The condition of turbine 4 is worse than of 3 and therefore they switched place in the sequence. Additionally, turbines 1 and 5 had better conditions than expected, so they were postponed by one day. Turbine 6 however, got worse and needed maintenance earlier than planned. Finally, on day is $t = 108$ the conditions of the remaining turbines are checked and rounded, and the turbines are placed accordingly:

```
schedule on day 108:  {105: [2], 106: [], 107: [3, 4, 6], 108: [1, 5]}
```

Turbines 1 and 5 are scheduled for maintenance on day $t = 108$ today and will therefore receive maintenance. It should be noted that turbine 5 is in good enough condition to wait 2 more days with maintenance. However, it is the last day in the period and the next period is in more than two weeks. Since this turbine is relatively close to the maintenance limit, it will receive maintenance now to reduce the risk of it failing.

D.3 Infeasible solutions from Approach 2

For this test run, $n = 32$ turbines are considered. The model from section 8.1 outputs a dictionary with *key=day* and *value=[list of scheduled turbines]*.

```
schedule: {105: [], 106: [], 107: [], 108: [13], 109: [26, 16, 8, 30],
    110: [1, 4, 15, 18], 111: [12, 14, 28, 2], 112: [25, 20, 17, 19, 29,
    23, 6, 7], 113: [24], 114: [9, 31], 115: [22, 5], 116: [10, 27, 3],
    117: [21], 118: [11, 32]}
```

The degradation paths for this result is presented in Table D.2.

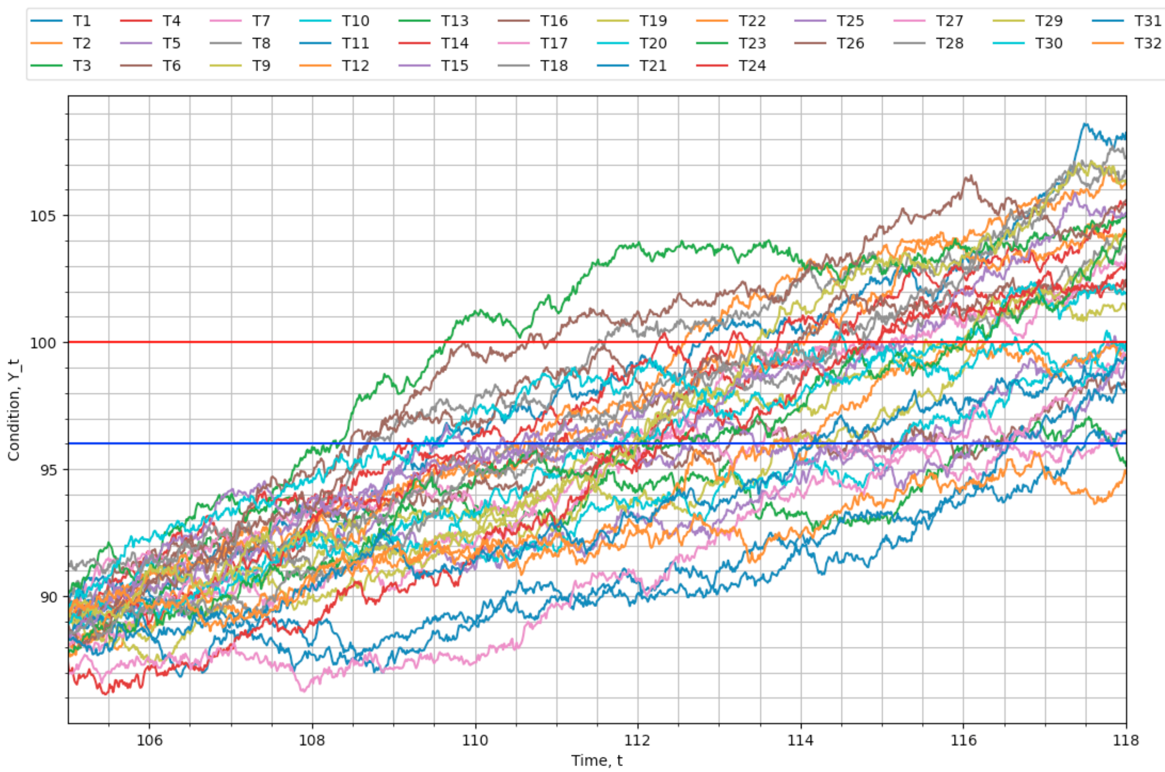


Figure D.2: Degradation paths for 32 turbines, T#, in first period from sub-model 1, Table 6.3

Comparison of results

To test how well the results of the proposed model came out, a simple and naive approach is set up for comparison. Starting at the first day of the maintenance window the *Naive approach*, fills the day up with the four turbines with the worst conditions. This procedure is repeated until there are no turbines left that require maintenance, which is after eight days. The results of six runs are presented in Table D.4. The results from the proposed model always come out better than *Naive approach*, but unfortunately these results are not sufficient since they only find the best time to do maintenance for the individual turbines and not the entire farm.

Table D.4: Comparison of results from proposed model to two relatively naive approaches

	Proposed model	Naive approach
Test 1: [cost]	332.69	349.13
Test 2: [cost]	332.61	346.86
Test 3: [cost]	331.90	346.41
Test 4: [cost]	332.03	346.13
Test 5: [cost]	331.94	347.06
Test 6: [cost]	332.13	345.52

Bibliography

Barros, A. (2019). Data driven prognostic and predictive maintenance. 1(2):1–59.

Bertsimas, D. and Tsitsiklis, J. N. (1997). *Introduction to linear optimization*, volume 6. Athena Scientific Belmont, MA.

Birge, J. R. and Louveaux, F. (2011). *Introduction to stochastic programming*. Springer Science & Business Media.

BSI (2010). BS EN 13306:2010: Maintenance-maintenance terminology. *British Standard Institution*.

CEN (2012). CEN - European Committee for Standardization.

Chen, Z. and Blaabjerg, F. (2009). Wind farm—a power source in future power systems. *Renewable and Sustainable Energy Reviews*, 13(6-7):1288–1300.

Chvatal, V., Chvatal, V., et al. (1983). *Linear programming*. Macmillan.

Cohen, D. (2004). *Precalculus: A Problems-Oriented Approach*. Brooks Cole, 6th edition.

Dai, L., Stålhane, M., and Utne, I. B. (2015). Routing and scheduling of maintenance fleet for offshore wind farms. *Wind Engineering*, 39(1):15–30.

Dantzig, G. (2016). *Linear programming and extensions*. Princeton university press.

Dantzig, G. B. and Wolfe, P. (1960). Decomposition principle for linear programs. *Operations research*, 8(1):101–111.

Deb, K. (2014). Multi-objective optimization. In *Search methodologies*, pages 403–449. Springer.

Durstewitz, M. and Wengler, R. (1998). Analyses of generator failure of wind turbines in germany. 250 mw wind programme. *Kassel: ISET*.

Equinor Renewables (2022). Data science problem statement, Dogger Bank scheduled maintenance optimization. *Restricted paper*.

Gao, R., Wang, L., Teti, R., Dornfeld, D., Kumara, S., Mori, M., and Helu, M. (2015). Cloud-enabled prognosis for manufacturing. *CIRP annals*, 64(2):749–772.

Gao, Z. and Liu, X. (2021). An overview on fault diagnosis, prognosis and resilient control for wind turbine systems. *Processes*, 9(2):300.

Gundegjerde, C., Halvorsen, I. B., Halvorsen-Weare, E. E., Hvattum, L. M., and Nonås, L. M. (2015). A stochastic fleet size and mix model for maintenance operations at offshore wind farms. *Transportation Research Part C: Emerging Technologies*, 52:74–92.

Gutierrez-Alcoba, A., Ortega, G., Hendrix, E. M., Halvorsen-Weare, E. E., and Haugland, D. (2017). A model for optimal fleet composition of vessels for offshore wind farm maintenance. *Procedia Computer Science*, 108:1512–1521.

Hahn, B., Durstewitz, M., and Rohrig, K. (2007). *Reliability of Wind Turbines*.

Hallowell, S. T., Arwade, S. R., Fontana, C. M., DeGroot, D. J., Aubeny, C. P., Diaz, B. D., Myers, A. T., and Landon, M. E. (2018). System reliability of floating offshore wind farms with multiline anchors. *Ocean Engineering*, 160:94–104.

Halvorsen-Weare, E. E., Norstad, I., Stålhane, M., and Nonås, L. M. (2017). A metaheuristic solution method for optimizing vessel fleet size and mix for maintenance operations at offshore wind farms under uncertainty. *Energy Procedia*, 137:531–538.

Hayes, G. (2019). Solving travelling salesperson problems with python. *Towards Data Science*.

Hofmann, M. (2011). A review of decision support models for offshore wind farms with an emphasis on operation and maintenance strategies. *Wind Engineering*, 35(1):1–15.

Igarashi, T. and Hamada, H. (1982). Studies on the vibration and sound of defective rolling bearings: First report: Vibration of ball bearings with one defect. *Bulletin of JSME*, 25(204):994–1001.

Irawan, C. A., Ouelhadj, D., Jones, D., Stålhane, M., and Sperstad, I. B. (2017). Optimisation of maintenance routing and scheduling for offshore wind farms. *European Journal of Operational Research*, 256(1):76–89.

ISO13372 (2012). Condition monitoring and diagnostics of machines — Vocabulary.

ISO13381-1 (2004). Condition monitoring and diagnostics of machines-prognostics part 1: General guidelines. *International Standards Organization*.

Jeon, S., Cho, Y., Seo, M., Cho, J., and Jeong, W. (2013). Dynamic response of floating substructure of spar-type offshore wind turbine with catenary mooring cables. *Ocean Engineering*, 72:356–364.

Kang, J., Sun, L., and Soares, C. G. (2019). Fault tree analysis of floating offshore wind turbines. *Renewable energy*, 133:1455–1467.

Karlõševa, A., Nõmmann, S., Nõmmann, T., Urbel-Piirsalu, E., Budziński, W., Czajkowski, M., and Hanley, N. (2016). Marine trade-offs: Comparing the benefits of off-shore wind farms and marine protected areas. *Energy Economics*, 55:127–134.

Kim, N.-H., An, D., and Choi, J.-H. (2017). Prognostics and health management of engineering systems. *Switzerland: Springer International Publishing*.

King, A. J. and Wallace, S. W. (2012). *Modeling with stochastic programming*. Springer Science & Business Media.

Land, A. H. and Doig, A. G. (1960). An automatic method for solving discrete programming problems. In *50 Years of Integer Programming 1958-2008*, pages 105–132. Springer.

Lien, L. C. (2021). Predictive maintenance for offshore wind farms.

Márquez, F. P. G., Tobias, A. M., Pérez, J. M. P., and Papaelias, M. (2012). Condition monitoring of wind turbines: Techniques and methods. *Renewable energy*, 46:169–178.

Mobley, R. K. (2002). *An introduction to predictive maintenance*. Elsevier.

Nemhauser, G. L. and Wolsey, L. A. (1988). *Integer programming and combinatorial optimization*, volume 191. Springer.

Peng, Y., Dong, M., and Zuo, M. J. (2010). Current status of machine prognostics in condition-based maintenance: a review. *The International Journal of Advanced Manufacturing Technology*, 50(1-4):297–313.

Pochet, Y. and Wolsey, L. A. (2006). *Production planning by mixed integer programming*. Springer Science & Business Media.

Rausand, M., Barros, A., and Høyland, A. (2021). *System reliability theory: models, statistical methods, and applications*, volume 396. John Wiley & Sons, 3rd edition.

Scheu, M., Matha, D., Hofmann, M., and Muskulus, M. (2012). Maintenance strategies for large offshore wind farms. *Energy Procedia*, 24:281–288.

Schrijver, A. (1998). *Theory of linear and integer programming*. John Wiley & Sons.

Schrotenboer, A. H., Ursavas, E., and Vis, I. F. (2020). Mixed integer programming models for planning maintenance at offshore wind farms under uncertainty. *Transportation Research Part C: Emerging Technologies*, 112:180–202.

Seyr, H. and Muskulus, M. (2019). Decision support models for operations and maintenance for offshore wind farms: a review. *Applied Sciences*, 9(2):278.

Shapiro, A., Dentcheva, D., and Ruszczynski, A. (2021). *Lectures on stochastic programming: modeling and theory*. SIAM.

Si, X.-S., Wang, W., Hu, C.-H., and Zhou, D.-H. (2011). Remaining useful life estimation–a review on the statistical data driven approaches. *European journal of operational research*, 213(1):1–14.

Sikorska, J., Hodkiewicz, M., and Ma, L. (2011). Prognostic modelling options for remaining useful life estimation by industry. *Mechanical Systems and Signal Processing*, 25(5):1803–1836.

Sinha, Y. and Steel, J. A. (2015). A progressive study into offshore wind farm maintenance optimisation using risk based failure analysis. *Renewable and Sustainable Energy Reviews*, 42:735–742.

Stålhane, M., Hvattum, L. M., and Skaar, V. (2015). Optimization of routing and scheduling of vessels to perform maintenance at offshore wind farms. *Energy Procedia*, 80:92–99.

Stålhane, M., Vefsnmo, H., Halvorsen-Weare, E. E., Hvattum, L. M., and Nonås, L. M. (2016). Vessel fleet optimization for maintenance operations at offshore wind farms under uncertainty. *Energy Procedia*, 94:357–366.

Tong, W. (2010). *Wind power generation and wind turbine design*. WIT press.

Vachtsevanos, G. J. and Vachtsevanos, G. J. (2006). *Intelligent fault diagnosis and prognosis for engineering systems*, volume 456. Wiley Online Library.

van Schrick, D. (1997). Remarks on terminology in the field of supervision, fault detection and diagnosis. *IFAC Proceedings Volumes*, 30(18):959–964.

Vatn, J. (2020a). Tpk4120 - lecture summary: Chapter 12 - preventive maintenance.

Vatn, J. (2020b). Tpk4161 - supply chain analytics.

Wang, M., Wang, C., Hnydiuk-Stefan, A., Feng, S., Atilla, I., and Li, Z. (2021). Recent progress on reliability analysis of offshore wind turbine support structures considering digital twin solutions. *Ocean Engineering*, 232:109168.

Wee, D., Kelly, R., Cattel, J., and Breunig, M. (2015). Industry 4.0-how to navigate digitization of the manufacturing sector. *McKinsey & Company*, 58:7–11.

Wu, X., Hu, Y., Li, Y., Yang, J., Duan, L., Wang, T., Adcock, T., Jiang, Z., Gao, Z., Lin, Z., et al. (2019). Foundations of offshore wind turbines: A review. *Renewable and Sustainable Energy Reviews*, 104:379–393.

Zhou, A., Yu, D., and Zhang, W. (2015). A research on intelligent fault diagnosis of wind turbines based on ontology and fmeca. *Advanced Engineering Informatics*, 29(1):115–125.

Ørsted (2019). Ever looked out from the cumbrian coast and wondered how an offshore wind farm works? everything you need to know about wind power (and were afraid to ask).

Lisa Chanel Lien

Combined models for optimization of routing and maintenance scheduling for offshore wind farms

NTNU
Norwegian University of
Science and Technology