Preventive maintenance optimization with updating capabilities for multi-component heterogeneous systems using multi-objective optimization

de Langen, B.W.H.

Award date:
2021

Department of Industrial Engineering and Innovation Sciences
Operations, Planning, Accounting and Control Group

Preventive maintenance optimization with updating capabilities for multi-component heterogeneous systems using multi-objective optimization

Master Thesis

Public version

B.W.H. (Bart) de Langen

Supervisors:
dr. C. Fecarotti, TU/e, OPAC
prof. dr. ir. I.J.B.F. Adan, TU/e, OPAC
dr. A.E. Akçay, TU/e, OPAC
O. Moers, Lely International

Eindhoven, September 2021

Eindhoven University of Technology, School of Industrial Engineering
Series Master Theses Operations Management and Logistics

Subject headings: maintenance policy, preventive maintenance, multi-component system, finite horizon, early exploitation phase of new systems, Bayesian updating, multi-objective optimization

It should be noted that all numbers costs are fictitious due to confidentiality reasons.

Abstract

The main objective of this research is to develop a multi-component maintenance concept for preventive maintenance for systems in an early exploitation phase. In this early exploitation phase, new data about the system and its component failure behaviour becomes available. A Bayesian updating method is applied to update the scale parameter of the Weibull failure distribution for Age-based maintenance (ABM) and Failure-based maintenance (FBM) components using event data. This event data is processed to get degradation data, which is used to determine the Gamma process parameters for Condition-based maintenance (CBM) components. Simulation models for all three (CBM, ABM and FBM) maintenance policies are developed to optimize the multi-objective component models on downtime and maintenance costs. These component models result in Pareto optimal component solutions that are combined with a Genetic Algorithm (GA) to find the Pareto optimal solutions at system level. A case study is performed on a high-tech system in the early exploitation phase used in dairy farming. In this case study, we show the results of a multi-component maintenance concept with Bayesian updating obtained by a GA.

Executive summary

This report is the result of a master thesis conducted at Lely International. Lely is an innovative company that develops and produces autonomic feeding, cleaning and milking systems for dairy cattle farms. These systems are maintained by service engineers from Lely Centers over the entire world. Lely Centers are the service and sales organizations that are mostly not owned by Lely.

Problem statement

Dairy farmers are depended on their systems for the continually running business of producing milk. System unavailability at a dairy farm is undesirable since it disrupts the business process and negatively impacts productivity. Besides that, an unavailable system could also cause health issues for cows such as cows that cannot be milked due to the breakdown of the milking robot. Altogether, this requires the high availability of the systems.

Developing a successful preventive maintenance concept for relatively new products is challenging since limited information is available on the machines' failure once the system is introduced to the market. This thesis follows up on the previous research of Hoedemakers (2020), in which a maintenance concept for new products is developed. At the start of a new system, the available failure information does not necessarily reflect the typical system failure. Therefore, we focus on the updating of a maintenance concept based on failure information from the field. This updating should result in a more realistic failure behavior of the system. Additionally, collecting the correct data contributes to the proper implementation of preventive maintenance. The component failure behavior degradation can be better determined when maintenance events are tracked during its lifecycle by collecting the correct information. Besides that, the results of a model are expected to be closer to reality when real data about failures is used. Everything together leads to the main research question to be answered in this research:

How to optimize a preventive maintenance concept for products in the early exploitation phase, based on field experience?

Research approach

This research is divided into four steps to answer the main research question with three objectives for Lely: minimizing downtime, costs and output consequences. The first step is finding a method to deal with Lely's available service data and give advice on how the collection of this data could be improved. The currently available service data can be grouped under Asset history data. This data contains information about the different maintenance events, such as Preventive maintenance (PM) or Corrective maintenance (CM) actions performed to systems. Other information is recommended to collect during the various maintenance events, such as the reason for a system breakdown during CM action. These error causes are currently not consistent and adequately collected. Collecting this information uniformly and adequately should result in data that gives a better understanding of the failure behaviour of systems. Additionally, the condition of components measured during PM actions is currently not stored. This information is helpful to determine the degradation path of components maintained by a Condition-based maintenance (CBM) policy.

The second step in this research is to find a multi-component maintenance model with a time horizon best suitable for Lely. This model should contain an updating feature to include information from newly collected service data. The maintenance model should be able to have Condition-based maintenance (CBM), Age-based maintenance (ABM) and Failure-based maintenance (FBM) as maintenance policies for individual components. A literature review concluded that a finite modeling horizon is the best horizon for a maintenance model at Lely. A finite horizon is preferred because of the requirement to update with newly available service data and replacements of components with new updated versions. Additionally, a finite horizon might be more appropriate when the expected lifetime of one or more components is close to the system's intended lifetime.

The models of Zhu (2015), Peng and Zhu (2017) and Shi et al. (2020) are selected as a basis for this research. The modelling approach of Zhu (2015) is used as starting point for our model. We changed the time horizon of this model and, therefore the solution approach as well. The system decomposition of Zhu (2015) is applied through simulations with a finite horizon to find system solutions. In the system, we used the maintenance intervals and maintenance thresholds as decision variables for the system and CBM and ABM components, respectively. The combination of a maintenance model and Bayesian updating is used from Shi et al. (2020).

Updating of failure distribution parameters is investigated in the third step of this research. The scale parameters of the Weibull distribution of ABM and FBM components are updated by a Bayesian updating method with the use of service data. For CBM, we determined the degradation parameters with the use of the service data. New failure parameters of the Gamma process are found by fitting a Gamma distribution on all available cleaned and processed service data. The fitted degradation parameters are also replaced in the maintenance model.

Results

Combining all information gathered in the first three research steps results in a multi-objective, multi-component maintenance model with updating capabilities. This model is applied in a case study on the Collector in the last step of this research. We found a Pareto frontier at the component level with Pareto optimal solutions between the two implemented objectives in this research: minimize maintenance costs and system downtime. Combinations of the component Pareto frontiers result in a possible system solution. Creating all different combinations resulted in an explosion of the solution space. Therefore, we first applied the Weighted Sum Method (WSM) to determine the system solution. Additionally, a Genetic Algorithm (GA) is used to find the Pareto optimal solutions at the system level for each maintenance interval τ. These Pareto optimal system solutions found by a GA includes the updating and replacements component parameters. The results are shown in Figure 1, where each '+' and '.' represent a system solution found by WSM and the GA, respectively. Figure 2 shows the Pareto optimal system solutions found by a GA.

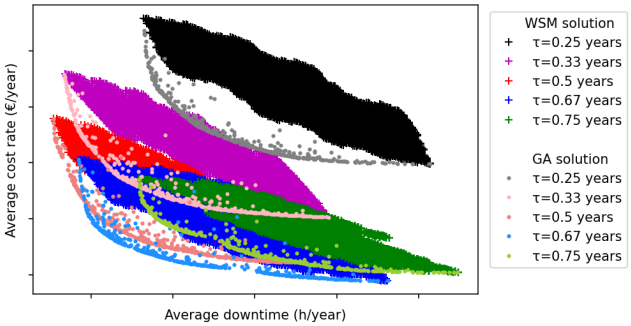

Figure 1: System solutions with updated parameter and different values of τ

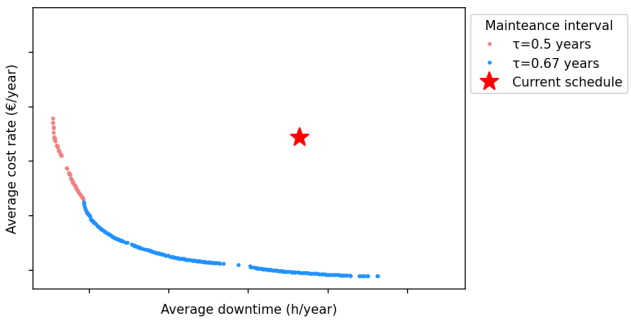

Figure 2: Pareto optimal system solutions compared with the current maintenance schedule

In Figure 1, we see different system solutions for multiple maintenance intervals τ. The crosses and dots with the same colour shade represent the same maintenance interval. This figure shows that the GA solution approach provides better system solutions than the WSM. The lighter coloured dots are mostly below the darker coloured crosses of the same maintenance interval. Figure 1 also shows that the maintenance interval of 0.67 years (8 months) gives a system solution with the lowest maintenance costs. The light blue dots also presents most of the Pareto optimal solutions at the system level. Only a limited number of solutions with a maintenance interval of 0.5 years (6 months) result in less downtime but higher costs. All Pareto optimal system solutions are shown in Figure 2. The currently applied maintenance schedule is also plotted in this figure. There can be concluded from Figure 2 that the current maintenance schedule can be improved in both costs and downtime regardless of the maintenance interval.

Recommendations

Update preventive maintenance schedules for new product

The first recommendation for Lely is to reconsider the preventive maintenance schedules and maintenance thresholds for relatively new products. We observed that the Weibull scale parameter could change by applying a Bayesian updating method with newly available service data. A change in this parameter intends that the failure behaviour of a component also changed. Incorporating these changes influence a maintenance model. Therefore, we recommend Lely update their preventive maintenance schedules for their new products, especially when they are just introduced to the market.

Service data collecting

Another recommendation for Lely is to collect their maintenance service data uniformly and consistently. Uniformly collecting maintenance data could be implemented by reducing the number of option categories or creating uniformly interpretable subcategories. In this way, most service engineers likely select the same options in the same situation. To collect this information consistently, reporting this information should be obligatory for some maintenance events, such as the reason for a breakdown during CM.

Predictive maintenance

The maintenance model which is created in this research focuses on Preventive maintenance (PM). The next step in maintenance is Predictive maintenance. This type of maintenance predicts the failure based on a component's historical data and its actual condition data. For the Collector, it is currently not possible to access the (condition) data remotely. For this reason, it is recommended to first access Collector data remotely with the help of IoT before starting with the Predictive maintenance step. The predictions of predictive maintenance can only be accurate when these are based on enough actual condition data; therefore, the condition data should be collected first.

Academic relevance

From an academic perspective, this research contributes by investigating the applicability of Bayesian updating of failure parameters in a multi-objective maintenance model. This model is based on the models of Zhu (2015), Peng and Zhu (2017) and Shi et al. (2020). The Bayesian updating method used actual service data to update the scale parameter. These updated parameters are implemented in a multi-objective maintenance model with a finite time horizon and static maintenance intervals. A Genetic Algorithm (GA) is applied to find Pareto optimal systems solutions based on the Pareto optimal components solutions. This model is used in a case study of high-tech equipment in an agricultural environment.

Preface

This report is not only the results of my Master Thesis project at Lely and the Eindhoven University of Technology. This report also marks the end of my student life. Completing my Master Thesis project would not have been possible without the help of others.

First of all, Claudia Fecarotti, my first supervisor from the university. I appreciate the feedback on the progress during our bi-weekly meeting and the advice on how to continue whenever I got stuck. Thank you for your time to help guide me through my master's project and provide me with insightful feedback.

Secondly, I want to thank Ivo Adan for being my second supervisor from the university. I really appreciate the feedback received during the progress meetings. Your feedback helped me to critically look at the approach of my project.

I also would like to thank Lely and Oscar Moers for letting me perform my research. Oscar, my supervisor from Lely, thank you for the guidance and for constantly helping me to see how things may be put into practice at Lely. Furthermore, I would like to thank Ipek Kivanç for the discussions and advice about modelling a maintenance model.

And finally, I would like to thank my dear friends and family for their support during my study and especially during this project. Above all, my girlfriend Claudia, thank you for supporting me with the delicious cakes you baked for me and the support you gave me through this project. You are the best!

Bart de Langen

Contents

List of Figures

List of Tables

List of Symbols

$\alpha_{i,HD}$	Scale parameter Weibull distribution of component i at update moment u from Hoedemakers (2020)
$\alpha_{i,u}$	Scale parameter Weibull distribution of component i at update moment u
$\beta_{i,HD}$	Shape parameter Weibull distribution of component i from Hoedemakers (2020)
β_i	Shape parameter Weibull distribution of component i
Δt	Time difference for degradation Gamma process
$\eta_{i,u}$	Scale parameter Gamma process of CBM component i at update moment u
$\gamma_{i,u}$	Shale parameter Gamma process of CBM component i at update moment u
ω	Weight of the WSM for each objective
ω_i	Weight of component i in the WSM optimization
τ	System maintenance interval for scheduled downs
τ^{LB}	Lower bound of the maintenance interval for scheduled downs
τ^{UB}	Upper bound of the maintenance interval for scheduled downs
Θ	Linear degradation rate of the RCM
A_i	Age limit of component i for component $i \in I_{ABM}$
b	The number of breakdowns in a simulation
$c_i^{additional}$	Costs of additional damage caused by component i
c_i^{CM}	Corrective maintenance costs of component i
$c^{engineer}$	Labor cost of a service engineer per hour
c_i^{part}	Costs of component i
c_i^{PM}	Preventive maintenance costs of component i
c_i^{REP}	Replacement costs of component i
c^{SD}	Fixed costs of a scheduled down
c^{USD}	Fixed costs of an unscheduled down
C_i	Control limit of component i for component $i \in I_{CBM}$
$cd_(x)$	Total crowding distance of solution x
cd_m	Crowding distance of objective m
$d_i^{additional}$	Downtime of additional damage caused by component i
d_i^{CM}	Corrective maintenance downtime of component i
d_i^{PM}	Preventive maintenance downtime of component i
d_i^{REP}	Replacement downtime of component i
d^{SD}	Fixed downtime of a scheduled down

d^{USD}	Fixed downtime of an unscheduled down
$D_{T_{i,n}}(\tau)$	Total downtime of component i with a certain maintenance threshold in simulation trace n and maintenance interval τ
F_i	Pareto front i with Pareto optimal solutions
F_m°	The ideal point of objective m
$F_m^{\max}$	The maximum value of objective m
$F_m^{\text{norm}}(x)$	The normalized objective function of solution x
f_x	The fitness of solutions x
H_i	Failure threshold of component i
I	Set of all components in the system $I_{CBM} \cup I_{ABM} \cup I_{FBM}$
I_{ABM}	Set of all ABM components
I_{CBM}	Set of all CBM components
I_{FBM}	Set of all FBM components
$K_{T_{i,n}}(\tau)$	Total costs of maintenance of component i with a certain maintenance threshold in simulation trace n and maintenance interval τ
l	Lifetime of a component in a simulation
o	The number of preventive maintenance actions in a simulation
p_i	The number Pareto optimal solutions of component i
s_x	The strength of solutions x
T	Finite time horizon of the system
$t^{drivingSD}$	Driving time to a farm for a scheduled down
$t^{drivingUSD}$	Driving time to a farm for an unscheduled down
t^{intake}	Duration for intake of an unscheduled down
t^{react}	Duration to react for a service engineer for an unscheduled down
$t_i^{replacement}$	Time for replacing component i
T_r	Number of simulation traces
t_u	Time in the time horizon of updating moment u
$V_{T_{i,n}}(.)$	The average yearly downtime of component i in simulation trace n and time horizon T
$V_{T_i}(.)$	The average yearly downtime of component i in time horizon time T
$V_{T_{sys}}(\tau)$	The average system down time per year at time T with maintenance interval τ
x_i	Initial degradation level of new CBM component i
$X_i(t)$	Degradation level of component i at time t
z_m^{max}	Maximal value of objective m
z_m^{min}	Minimal value of objective m
$Z_{T_{i,n}}(.)$	The average yearly costs of component i in simulation trace n and time horizon T
$Z_{T_i}(.)$	The average yearly costs of component i in time horizon T
$Z_{T_{sys}}(\tau)$	The average system costs per year at time T with maintenance interval τ

List of Abbreviations

Abbreviation	Description
ABM	Age-based maintenance
CBM	Condition-based maintenance
CM	Corrective maintenance
CTBM	Calendar time-based maintenance
FBM	Failure-based maintenance
GA	Genetic Algorithm
MLE	Maximum Likelihood Estimation
MOGA	Multi-Objective Genetic Algorithm
MOOP	Multi-objective optimization problem
NSGA	Nondominated Sorting Genetic Algorithm
NSGA-II	Fast Nondominated Sorting Genetic Algorithm
PdM	Predictive maintenance
PM	Preventive maintenance
RCM	Random Coefficient Model
RWGA	Random Weighed Genetic Algorithm
SD	Scheduled down
SPEA	Strength Pareto Evolutionary Algorithm
TSS	Technical Service Support
UBM	Usage-based maintenance
USBM	Usage severity-based maintenance
USD	Unscheduled down
WSM	Weighted Sum Method

Chapter 1

Introduction

The first chapter gives an overview of the context where this research is carried out. This chapter provides background information about the company, the project and terminology used in this research. Furthermore, the considered system in this research and its current maintenance schedule are given in this chapter.

1.1 Maintenance terminology

This section provides a basic overview of maintenance and the maintenance terminology that is used in this report. In general, most authors in the maintenance literature agree on the definition of maintenance as a *"set of activities necessary to keep physical assets in the desired operating condition or to restore them to this condition"* (Pintelon and Parodi-Herz, 2008). Additionally, there is more confusion regarding the meaning of maintenance actions, maintenance policies and maintenance concepts. The definitions of these terms from Pintelon and Parodi-Herz (2008) are used in this report. A maintenance action is an elementary task, such as an inspection or repair, executed by a service engineer. A maintenance policy is the (set of) rules which triggers a maintenance action. A maintenance policy is used to clarify the maintenance type of a single component. A maintenance concept is a set of actions and policies that describe how maintenance is carried out on a system. Altogether, the term policy is used for single components and a concept for a complete system.

There are different maintenance policies described in the literature. In this report, a combination of maintenance policy terminology from Tinga (2010) and Arts (2017) is used. Maintenance policies in this research are classified into modificative, preventive, predictive and breakdown corrective maintenance, as shown in Figure 1.1. The dashed region in Figure 1.1 covers maintenance policies that can be used in a predictive maintenance policy.

A modificative maintenance policy involves interchanging a component with a new, more advanced component. This maintenance policy is usually project-based and thus non-recurring. Modificative maintenance occurs at Lely when a system component is redesigned for an existing product due to quality or safety issues of the component. The Corrective maintenance (CM) policy implies components are only repaired or replaced after a failure. Breakdown corrective maintenance can be a suitable option not wearing components whose failures are not critical. A Preventive maintenance (PM) policy aims to replace components preventively to avoid failures. When a component breaks down before the scheduled PM action a corrective maintenance action is usually performed.

PM policies can be further divided into Usage-based maintenance (UBM) and Condition-based maintenance (CBM) policies. Under UBM, a component is replaced preventively after a predetermined usage threshold is reached. The component usage can be further divided into Calendar

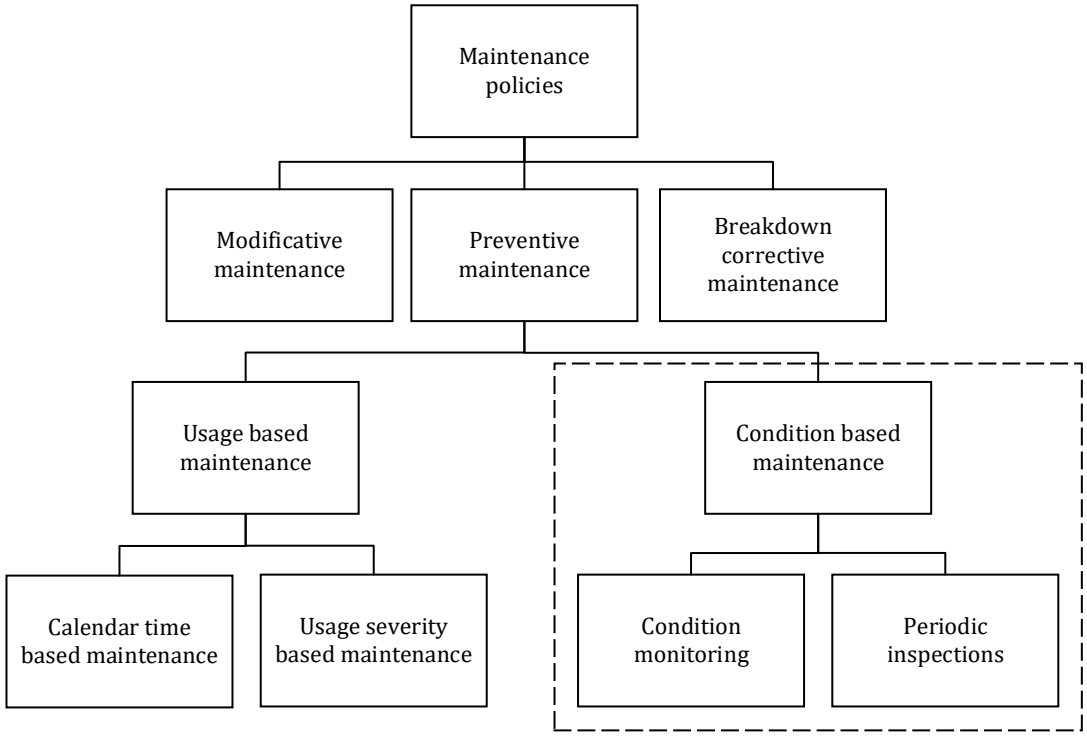

Figure 1.1: Maintenance landscape with predictive maintenance policies in the dashed region, redrawn from Arts (2017)

time-based maintenance (CTBM) and Usage severity-based maintenance (USBM). CTBM is the easiest way of defining component maintenance intervals by using a fixed period, such as every month or every year. Age-based maintenance (ABM) is another term used for CTBM. The ABM maintenance policy applies when a component degrades due to aging, like exposure to UV radiation. USBM takes the actual usage of an operating system into account and not an assumption on the usage severity of the system during operating hours. This maintenance policy can only be applied when the relation between usage severity and life consumption is known. Besides that, usage monitoring like mileage driven, times charged, or performance indicators are necessary to collect next to operating hours for USBM.

In CBM, the maintenance actions are based on the conditions of a component. The component condition can be measured directly by the actual condition or by measuring parameters related to the component condition. Both measuring methods can be performed during visual inspections or continuously by using sensors. More advanced sensor technology and developments in the IoT made continuous monitoring via direct and parameter measurements a lot easier. Additionally, technological advancements made continuous condition monitoring possible nowadays. Sensors became less expensive with higher accuracy levels of measuring temperatures of liquids and amplitude of vibrations. Maintenance activities may be scheduled closer to component failure periods when using real component conditions, which is an advantage of CBM over UBM. However, continuous condition monitoring is not always possible and can be very expensive. Therefore, the cost-effectiveness of condition monitoring should be assessed. This should be based on the criticality of the system component concerning the system function, performance and the maintenance costs.

Sensor data from continuous monitoring provides more information on component conditions over

time than visual inspections. CBM only uses the current component condition for a maintenance decision. Maintenance should only be performed when a component condition exceeds a control limit. Although Predictive maintenance (PdM) is often referred to as CBM, PdM goes further than CBM. Predictive maintenance uses historical data and component conditions to predict the occurrence of a failure under normal conditions (Shafiee, 2015). Analyzing historical component condition data can result in a degradation path that can be used to predict the expected remaining useful life of a component (Van Horenbeek and Pintelon, 2013). The combination of real-time condition information and the component degradation path in PdM makes it possible to schedule maintenance closer to the predicted moment of failure, which increases the life span of a component.

1.2 Company background

This research is conducted at Lely, an agricultural family-owned company founded in 1948 by the two brothers' Cornelis and Arij van der Lely. Lely's headquarters are located in the Netherlands in Maassluis, the same town where they were founded in 1948. The brothers' Van der Lely worked on innovations to make the life of farmers more pleasant. Their first innovative product was a rake used for collection mowed grass. Ten years later, Lely developed a fertilizer spreader that is still produced and sold in America nowadays. The introduction of the power harrow in 1968 resulted in a significant sales increase and the international breakthrough of Lely. In 1992 Lely introduced the first generation of their automatic milking system, the Astronaut. The automated milking system is seen as the biggest innovation for dairy farming in the 20th century by dairy farmers. Currently, over 30,000 milking robots are sold worldwide over five generations of the Astronaut. This makes the Astronaut still one of Lely's most essential products. In 2017, Lely made a strategic decision to focus on dairy farming worldwide and the sale of their forage activities.

Globally Lely has approximately 1,600 employees, of which more than 800 employees are working at the Lely campus in Maasluis. For organizational purposes, Lely has customers in more than 45 countries grouped into 11 clusters (e.g., Benelux, Oceania). Each cluster is a group of mainly privately owned Lely Centers in a specific region. These Lely Centers are the sales and service organizations that have close contact with their customers. Each Lely Center has its service engineers, inventory and customers for which they perform maintenance. Lely makes maintenance schedules that Lely Centers can use for performing maintenance for their customers. All Lely Centers schedule and perform maintenance themselves for their customers. Lely's total sales revenue in 2020 was €615 million, of which 6% is invested in the three R&D departments of Lely. Their 1,600 active patents can illustrate the innovation of Lely in 2016.

1.3 Project context

This project is performed at the Competence Center Service Technologies within Lely. This is a new center that aims to develop the field service methodologies. There is a close link between this center and Lely's Technical Service Support (TSS) department. The competence center focuses on developing knowledge and methods of new service concepts, whereas TSS focuses on an operational basis. TSS addresses technical malfunctions of Lely products at farms, resulting in a lot of practical knowledge available at TSS.

Lely is an innovative company with a focus on sales of their innovative products. Besides the sales of products, Lely also offers certain maintenance services. For several years Lely has aimed to become a more service-orientated company. The road map that Lely should guide towards a more service-orientated company is named Lely's servitization journey. Servitization was first used in 1988 by Vandermerwe and Rada as a customer-focused concept by combining products, services, support and knowledge as its most essential elements. The four phases of Lely's servitization journey are named based by Lely's relationship with the customer. The relationship changes from

Lely as wholegood supplier in the first phase, via the service provider and trusted advisor in the second and third phase, to strategic business partner in the fourth phase. In all these four phases, Lely's revenue model, maintenance concept and customer support will change.

This research focuses on the changes in a maintenance concept in Lely's servitization journey. The maintenance concept changes in the four phases from breakdown corrective maintenance in the first phase via preventive and predictive maintenance to pro-active maintenance in the last stage. Lely's servitization journey's final goal is to offer 100% uptime of Lely products without unscheduled breakdowns and with a limited number of service visits scheduled at customers. This thesis covers the last step of the second phase in Lely's road map to servitization: updating opportunities in a preventive maintenance concept.

1.4 Project description

Developing a preventive maintenance concept requires knowledge about component degradation, component failure behaviour and the failures' impact on the system. However, there is usually not any field data available for new products. The reliability information of products and their components can be tested via accelerated lifetime tests during product development. These tests provide the first failure behaviour insights of a system. Nonetheless, there are multiple limitations to this gathered data due the test to the environment, duration and costs. Therefore, creating a maintenance concept based on test data could result in high maintenance costs or a high number of unplanned breakdowns.

Currently, there is a maintenance model available with preventive maintenance policies to find a maintenance concept for new products at Lely. This available model was developed by Hoedemakers (2020) in previous research. The input parameters of the failure behaviour in this model are mainly based on data gathered from testing and expert knowledge. The next step is adapting this model to make it suitable for products in the early exploitation phase. More field data is collected in the early exploitation phase of a product, resulting in more reliable data about product failures. There are three maintenance alternatives to investigate with more reliable maintenance data: a static but robust approach, a static more risk-prone approach, or an updating approach. These three alternatives can have their advantages as well as their disadvantages.

First, a static and robust preventive maintenance approach. This approach implies that components are replaced before failures can occur. An advantage of a robust approach is to reduce the number of unscheduled downs of a system. However, the first disadvantage of a robust approach is the premature replacement of system components that discard useful component lifetime. Additionally, it is not possible to collect new field failure data of these components when they are replaced before they fail. The disadvantages of the static but robust approach outweigh the advantages in Lely's path toward servitization.

A risk-prone maintenance approach allows for failures in the system, which is a type of breakdown corrective maintenance. A risk-prone approach enables the desired failure data to be collected from the system. However, these failures cause a lot of unwanted downtime from the system. Customers who have purchased a system expect a product with high quality and availability. As a result, this approach is unlikely to be accepted by customers. Therefore Lely does not favour the risk pone approach either.

A third approach is an intelligent approach were newly collected maintenance data is used to update and optimize the maintenance approach. In this way, helpful knowledge can be extrapolated from the collected data by analysis. The information gathered from the collected data should be used for optimizing a maintenance policy. Updating a maintenance policy with newly collected data probably results in an improved static maintenance concept. This updating approach is a

promising method to arrive at the proper maintenance concept for a system.

All three mentioned approaches are methods to deal with newly collected data in maintenance. The third approach is the most promising method of including new field data into a maintenance concept. This approach is next to a promising method for the most complex procedure. Decision rules should be defined to determine if and when the maintenance concept should be updated. These decision rules should prevent updating the concept and thresholds after every new observation. An updating decision rule could be: when the difference between a calculated threshold in the concept and the current threshold derived from data deviates more than a predetermined level, the threshold in the concept should be updated. Furthermore, rules should be determined about how thresholds should be updated. The weighted average of the old threshold from the concept and the current threshold of the data is an example of determining the new threshold.

1.5 Discovery Collector

The product considered during the case study of this research is Lely's Discovery Collector 120, which is referred to as Collector in this report. Figure 1.2 shows pictures of the Collector at a barn [1]. The Collector was introduced in 2016 and is a relatively new product. Currently, more than 1500 Collectors are sold. The introduction year and amount of sold systems make the Collector suitable validating a maintenance concept for products in an early exploitation phase. The Collector is a self-driving robot that takes the manure away in barns with solid floors by suction.

Figure 1.2: The Lely Discovery Collector 120

Before the Collector was introduced, the alternative which many farmers used was a mechanical manure scrapper. Farmers periodically used this to clean the barn. The manure scrapper was connected with a metal chain to a motor. This motor pulled the manure scrapper over the floor, which pushed the manure to the end of the barn. The manure scrapper is an easy and cheap system to clean the cowshed but also has limitations. Firstly, the metal chain between the manure scrapper and motor restricts cows' movement in the barn. Secondly, the scrapper is not able to reach all corners and ends due to the inflexible chain. Finally, pushing the manure with the scrapper results in a thick layer of manure that the cows in front of the stand in with their hoofs. This is not hygienic and could cause health issues for the cows. The Collector is the solution for all limitations of the mechanical manure scrapper.

The Collector operates as follows: The Collector receives a day planning to clean the barn, consisting of several programmed cleaning routes. All these cleaning routes start at the charging station with an empty manure tank. The water bags of the Collector are filled with water at the water station before driving a cleaning route. During a cleaning route, the Collector navigates itself through the barn using a gyroscope, two ultrasonic sensors, the number of wheels rotations and

[1] Retrieved from `http://www.lely.com`

the programmed map of the barn. The Collector uses the wall guides to protect the system and to drive along walls. During a cleaning route water is pumped out of the water bags and is sprayed at the front and back of the robot by two nozzles. The water sprayed in front of the Collector dilutes the manure, which makes it easier to collect. The water sprayed at the backside makes the floor less slippery for the cows. A vacuum pump inside the Collector is used to create suction to collect the manure in the barn. The scrappers and floor guides of the system together guide the manure to the manure tank in the Collector. A top-level sensor measures whether the manure reaches the maximum level in the tank. When this level is reached, the vacuum and water pump are turned off. The Collector continues his route to the dump station where the aeration valve is opened, the vacuum disappears and the manure is dumped at the dump station. The empty Collector returns to the charging station to recharge the battery for the following route.

1.6 Current maintenance

Since this thesis focuses on a preventive maintenance concept that will be tested on the Collector, an introduction about the current maintenance of this machine is given. Currently, Lely uses two maintenance policies: preventive and Corrective maintenance (CM). The preventive maintenance policies used are Condition-based maintenance (CBM) with periodic inspections and Age-based maintenance (ABM).

When Lely launched the Collector, there was no maintenance concept available that specifies which maintenance actions should be performed at which moment. Due to several deficits at the Collector during the product launch, many corrective maintenance was performed in the first year. After this period, several components are redesigned to increase the reliability of the system. Besides that TSS defined a maintenance schedule with preventive maintenance actions to reduce the number of unscheduled downs.

The currently used maintenance schedule consists of 3 types of preventive maintenance visits (A, B and C). During a particular preventive maintenance visit, a service engineer should perform a list of predefined actions, such as cleaning, inspecting and replacing components. The three different preventive maintenance visits are performed in the A-B-A-B-C sequence. The interval between these preventive maintenance visits is set at six months. This sequence of preventive maintenance visits is repeated every three years. Furthermore, breakdown corrective maintenance is performed when the system breaks down between two preventive maintenance visits. The opportunity of performing preventive maintenance during a corrective maintenance visit is a maintenance opportunity. The use of maintenance opportunities is outside this project's scope because another Lely department investigates this.

Besides the corrective and preventive maintenance actions of service engineers, Lely defines operator maintenance. Operator maintenance are minor maintenance actions executed by users of Lely products. Cleaning ultrasonic sensors, fixing congestion at the water station and cleaning the air filter are examples of these minor maintenance actions. Performing these minor actions periodically should result in fewer corrective maintenance visits. There is assumed that Lely customers perform all prescribed operator maintenance actions correctly. Only the maintenance executed by Lely's service engineers is investigated during this project.

Lely's servitization journey toward a more service-oriented company requires changes in its current maintenance approach. The journey's final goal is to offer 100% uptime of Lely products without unscheduled breakdowns and a limited number of service visits to customers. Hoedemakers (2020) started this journey by creating a model with preventive maintenance policies to create a maintenance concept for new products. This research adapts Hoedemakers' model to make it suitable for products in the early exploitation phase by adding updating opportunities of newly collected data.

Chapter 2

Research Proposal

This research aims to develop a tool that creates maintenance concepts for products in the early exploitation phase. This Python tool includes a modelling framework for developing and optimizing maintenance concepts. By using this tool, Lely can develop maintenance concepts for their products. The model used in this tool is demonstrated and implemented to the Collector. This chapter presents the problem statement, the scope and the research questions with their approach.

2.1 Problem statement

Dairy farming is a continuous business where farmers depend on their systems for daily operations. Unavailability of systems at a dairy farm results in disruption of business processes and less productivity. Besides that, an unavailable system could also result in health issues for cows; for instance, a cow cannot be milked due to a long breakdown of the milking robot. Altogether, this ensures that farmers require the high availability of their systems.

Proper implementation of preventive maintenance contributes to the higher availability of systems. Developing a successful preventive maintenance concept for relatively new products is challenging because little information is available on the machines' failure behaviour from the field. The available failure information does not necessarily reflect the typical failure of system components. Therefore updating of component failure distributions with new field service data should result in more reliable failure behaviour. Additionally, collecting the correct data could also contribute to the proper implementation of a maintenance model. By collecting the correct information, the component failure behaviour can better be determined. The use of reliable component failure information should ensure that the model results are more reliable and closer to reality.

2.2 Project scope

This research focuses on the last step of the second phase in Lely's four phases road map to servitization. The four phases in this road map correspond with different maintenance concepts: corrective maintenance in the first phase, preventive maintenance, predictive maintenance in the second and third phase and pro-active maintenance as the final goal. In a previous master thesis, Hoedemakers (2020) finished the first part of the preventive maintenance phase by developing a framework for a maintenance concept for new systems. This research follows up on the same topic, focusing on updating of a maintenance concept based on newly available field data. Besides that, this study also concentrates on data collection and processing data used for maintenance purposes.

The outcome of this thesis should be a modelling framework to develop maintenance concepts for new systems during the early exploitation phase. The framework shall have the capability to model a multi-component system and enable updating decisions based on new field service data.

Furthermore, this framework should be flexible enough to be applied to a variety of Lely's products. Updating the maintenance concept should reduce the uncertainty of lifetime distributions and degradation patterns of a system. There is no geographical distinction made in the model and the data used for updating. However, the model and updating method should be flexible in dealing with, for example, geographical distinctions for the final goal: an individualized pro-active maintenance concept.

Preventive maintenance intervals of system components will be aligned to the maintenance intervals at system levels. Performing preventive maintenance on the systems during a system failure between two preventive maintenance visits is not considered in this thesis because another department within Lely investigates this.

This research is focused on the maintenance performed by service engineers of the Lely Centers. The routine of minor maintenance actions executed by operators of the machines is out of this project scope. It is assumed that these little maintenance actions are performed as prescribed by Lely.

Finally, the maintenance planning horizon considered in this research is the finite horizon. The resource planning on an operational level is not within the scope of this research. There is assumed that sufficient resources will always be available to perform maintenance actions.

2.3 Research goal and deliverables

This master thesis project focuses on preventive maintenance for products in the early exploitation phase using component reliability updating. Before implementing preventive maintenance, it has to be investigated when a preventive maintenance policy would be suitable and whether it would be the most beneficial option. Preventive maintenance can be divided into usage and condition-based maintenance policy. The failure rates of components should be determined from data in order to choose the best maintenance policy for them. The available data will be processed to find information about component failure behaviour. Additionally, there will be a closer look at collecting data for maintenance purposes. Maintenance models with different time horizons will be analyzed to find the best suitable time horizon for Lely. An updating procedure in the maintenance model of component reliability parameters allows us to include expert knowledge, test data and newly collected service data.

This research aims to develop a framework for a maintenance concept for products in the early exploitation phase. Corrective, usage-based and condition-based maintenance policies are considered for all the products components. Furthermore, the research outcome has to be applicable to a variety of Lely's products. Additionally, the theoretical framework is implemented with a case study on the Collector.

By successfully conducting this research, we make the following contributions to Lely:

1. Recommendations on which field service data is recommended to collect for continuous improvement of maintenance concepts and product development
2. Guidelines on how to collect service data uniformly
3. Recommendations about different time horizons in multi-component maintenance concepts
4. Defined rules about updating a maintenance concept with new field service data
5. Implementation of an update mechanism to the maintenance concept in a Python model
6. Validation of maintenance updating mechanism on the Collector

2.4 Research questions and approach

To reach the research goal and deliverables, we define the following main research question:

How to optimize a preventive maintenance concept for products in the early exploitation phase, based on field experience?

Several sub-questions are defined to answer the primary research question. The methodologies used to answer these sub-questions are discussed for each research question.

1. **How to deal with data collection from products for maintenance purposes?**
    i. *What service data should be collected for optimizing a maintenance concept?*
    ii. *How to deal with right-censored, incomplete and incorrect collected service data?*

The first research question involves the first two steps of Condition-based maintenance (CBM) according to Jardine et al. (2006): data acquisition and data processing. CBM is the maintenance policy that requires collecting the most data compared to corrective and usage-based maintenance. We have to understand what type of data Lely should collect that represents the health of the system and its components. Besides CBM, there are also system components for which corrective or usage-based maintenance is the best maintenance policy. Different data is likely desired to be collected for other maintenance policies. First, a literature review will be conducted on how to collect service data for maintenance purposes. Additionally, we have to develop a method to deal with incomplete, censored and unreliable field data. After this, we describe which data is available from Collector and how to deal with this data.

2. **Which maintenance optimization model is best suited to develop a maintenance concept for Lely's machines, and which planning horizon assumptions best fit Lely's features and requirements?**

In the second research question, we investigate three different time horizons in a multi-component maintenance model. A model created in previous research at Lely uses an infinite time horizon. However, Lely produces systems with a finite life cycle. Therefore, a literature study will be conducted to find and highlight the differences between the effects of these time horizons. As a result of this research question, the time horizon that best matches the purposes of Lely will be implemented.

3. **How to include updating of components failure information based on new evidence in a maintenance concept?**
    i. *How to perform updating with new component failure/condition information?*
    ii. *When should a maintenance concept be updated?*

This research question aims to create a preventive maintenance concept for products in the early exploitation phase. For products in this phase, more operational and maintenance data from the field becomes available. Therefore, updating component reliability parameters based on new product data in the maintenance model will result in an iterative model. A literature review about updating procedure and the updating moment of a maintenance model will be conducted to answer this research question. The most suitable method for an iterative updating of a maintenance model will be implemented to a multi-component maintenance model. The updating should probably result in earlier replacements of a frequently failed component while components that (almost) never fail before their replacement may be replaced later.

4. **How can the framework be applied to the Lely Collector?**

    i. *How to discover when too much or too little maintenance is carried out and adapt this to the maintenance model?*

Using the iterative updating maintenance framework in practice is applied in a case study on the Collector. The parameters of Hoedemakers (2020) model are used as a starting point during this case study. These parameters are updated based on newly collected data. The updating of the parameters in the maintenance model will likely result in different maintenance thresholds for individual components and maintenance intervals. According to the model, a result of the case study based on service data is a list of components that are replaced too early or too late.

2.5 Requirements model

This section discusses Lely's requirements for a maintenance model for this study. These model requirements are used as starting point for the literature research with a multi-component maintenance model with an updating procedure. The characteristics of the maintenance models are classified and filled in according to Lely's requirements in this section.

There are several characteristics to classify multi-item maintenance models in the literature. Different classifications of multi-item maintenance models are used in three frequently cited review articles. Cho and Parlar (1991) divide multi-component models into five categories: repair models, opportunistic models, inventory models, other replacement models and inspection models. Nicolai and Dekker (2008) categorize based on component dependencies, planning horizon and optimization methods. A literature review by de Jonge and Scarf (2019) distinguishes single and multi-unit systems. They classify these models based on replacements, repairs, inspections and the deterioration process. These papers are used to create categories of maintenance models.

The desired maintenance environment for Lely in this study is described in Table 2.1, which is explained further in this section. This study is a step in Lely's servitization journey to obtain an individualized pro-active maintenance concept. The model at the end of the servitization journey may have other requirements than described in this section because this study is a step forwards to the goal of the servitization journey.

Table 2.1: Model requirements

Number of components	Individual policies	Minimal repair	Maintenance interval	Failure interaction	PM opportunities	Updateable
100 +	FBM, UBM & CBM	No	Static	Positive economic	Only at SD	Parameter updating

After several discussions with Lely, it became clear that Lely requires a general model that should apply to all Lely's systems. This includes systems that are currently produced as well as systems that will be produced in the future. Therefore Lely wants a generic maintenance model. This generic model should not assume specific degradation patterns or lifetime distributions and should be applicable to all systems. The complex systems of Lely have more than a hundred components. Therefore the model should be able to deal with a large number of components.

Lely produces complex systems with components that are subject to failures. These components are non-identical, which means that each component has its individual best maintenance policy. Therefore, the model should consider many components maintained with different policies such as FBM, ABM and CBM.

Lely's service engineers use mainly new parts during maintenance visits. Minimal and imperfect repairs are therefore not included in the model. The model should only have perfect repair actions during maintenance visits. These maintenance visits should always have the same static maintenance interval. It is also allowed to change the maintenance interval from one static interval to another static interval. This makes the model realistic for changes in maintenance schedules but also convenient to implement at this moment.

The set-up cost for maintenance actions at Lely covers a substantial part of the total maintenance costs. These costs can be divided over multiple maintained components during a planned maintenance activity. The spread of these set-up costs leads to significant cost savings compared to maintaining all components individually. Therefore, the positive economic dependencies of components are the highest priority of the model.

The maintenance model should only consider PM action during Scheduled down (SD). Currently, only the failed components are replaced during a system breakdown or Unscheduled down (USD). Therefore, the model should only include PM during SD. In the future, the model could be expanded with PM actions during USD.

In the early exploitation phase, more data of the system and components becomes available. The newly available data from component replacements can provide new insights into component lifetime behaviour. Lely wants to include these new insights in its maintenance model. This can be achieved by updating the component failure parameters with the newly available data. The updating of the parameters can results in changeable maintenance rules in the model planning horizon.

Chapter 3

Data acquisition and processing

In this chapter, the first research question is answered:

1. *How to deal with data collection from products for maintenance purposes?*

This chapter is divided into three sections on extracting information from data to answer this research question. The first section explains the different types of maintenance-related data. The second section concerns data collection and specifies the available maintenance data at Lely. This section also provides advice on which maintenance-related data is relevant to collect in the future for Lely. The third section describes the data processing steps performed on the available information.

3.1 Data types

In this section, we describe the maintenance-related data types according to the literature. There is a variety of data that can be collected during and after maintenance activities. All this data can roughly be categorized into two main types according to Jardine et al. (2006): event data and condition data. Event data include information on what happened and what is performed to a system. Examples of what happened to a complete system are installations and breakdowns. The second part of event data includes information on a system's performance, such as preventive maintenance and minor repairs. The second category is condition data, which contains information related to the condition of an asset. Condition data can be collected in two ways, by condition monitoring and during a periodic inspection. Condition monitoring is the continuous measurement of a component condition or parameters related to a component condition by sensors. During a periodic inspection, the condition of a component can be determined in two different ways: by measuring a condition or assessing a component's condition. Both are examples of periodic condition measurement of the component at certain moments.

Tiddens (2018) distinguishes maintenance data into four categories: asset history data, usage data, stressor data and condition/health data. Figure 3.1 gives a schematic overview of these categories. The first category, asset history data, covers technical knowledge, inspection data and system failure records. This category contains most of the event data category that Jardine et al. also described. The second category is usage data. The data in this category consists of usage and process data, such as running hours or tons produced. Stressor data is covered in the third category. Data in this category contain information about environmental and exerted loads (stressors) on a system. Loading data includes information about the load on components such as vibrations or electric current, whereas moisture and temperature are examples of environmental data. Condition/health data is the last category that contains condition, test and health data. Condition data includes information on the component condition or parameters related to this condition. Health monitoring data includes information on measured vibrations in structures to

identify damages. The test data contains information gathered from accelerated lifetime tests during the development of a system or component (Tinga, 2010).

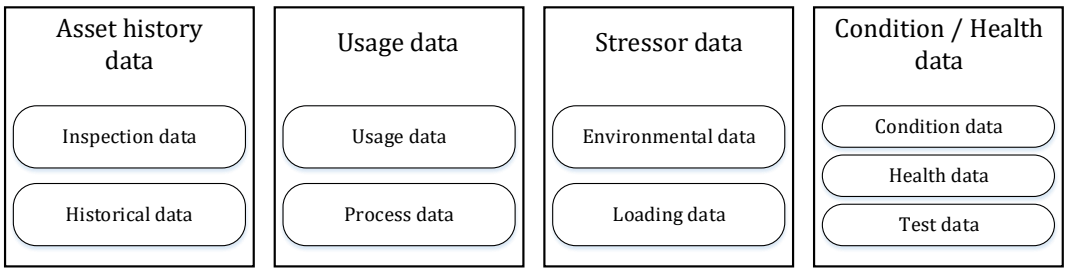

Figure 3.1: Maintenance data types, redrawn from Tiddens (2018)

The four data categories of Tiddens are more detailed than the two categories that Jardine et al. describe. Therefore, the data categories of Tiddens are therefore used to categorize the available data of the Collector in the next section.

3.2 Data collection

This section is divided into three sub-sections. Firstly, Lely's available maintenance data is distributed among the data categories of Tiddens (2018). Secondly, there is looked into the literature for relevant maintenance service data. The knowledge from the literature and the currently available data at Lely is combined to give advise on which service data is recommended to collect for maintenance purposes in the future.

Available data

Tiddens four data categories are used to categorize the maintenance data. First, we look at which data is available in these categories for the Collector at Lely. The service data of the Collector belongs to the asset history data. This data contains information about maintenance events performed at a machine by Lely service technicians. There is currently no usage data from the Collector available. This data could be obtained via calculation with available data saved in the Collector log files. The online availability of these log files via IoT would be a convenient way to collect usage data. An example of stressor data at the Collector is kilograms of manure collected. This information is unfortunately not measured and saved for the Collector. Only the moments with a full manure tank are saved in the log files. Ambient temperature is an example of environmental information that the Collector also saves in the log files.

The last category is the health/condition data. The Collector creates log data that could show relations between information measured by sensors and a component condition. The resistance, friction, voltage and current are examples of this information. The sensor data is saved in the same log file of the Collector for at most two hours. These two hours consist of the last full hour and the number of minutes in the current hour. For example, the Collector crashes at 13:47; then all sensor data is available from 12:00 until 13:47. When the Collector crashes at 13:01, then all censor data is available from 13:00 until 13:01. Lely is rolling out the cloud storage of the log files that are currently not available. Therefore, it is currently not possible to establish links between the values in this data and component conditions.

In this research, we mainly use the service data which falls under the asset history category. The available service data contains information about the moment of system installations, the type of performed maintenance (e.g., corrective or preventive maintenance), the parts used, costs made,

and the date of a service event at a specific machine. Besides that, information about the machine run time in days, error descriptions and error symptoms is available in this data. The current error descriptions contain information that describes the situation during a repair which a service engineer selects from several categories. These error descriptions are not always filled in consistently. Besides that, the error symptom information that provides more detailed information about the error and the repair is manually typed text in different languages. Both make this data hard to use at this moment.

It is difficult to extract information from a data set in which specific information can hardly be analysed. Especially when sections of the data are entered inconsistently in different languages. Fortunately, the data also contains information on which parts are replaced on individual machines and what dates. This data includes relevant information on the lifespan of Collector components when processed in a certain way. Before processing the data, it is important to filter out incomplete or less reliable data. These steps of data cleaning and data preparation are further discussed in section 3.3.

Maintenance data collection in the literature

Properly collected error symptoms and descriptions can be helpful information for uncovering failure root causes or predicting component failures. Both error-related information is event data and could therefore be improved in Lely's service data. Error causes are categorized into seven cause categories, according to Geitner and Bloch (2012). The "Seven Root-Cause Category Approach" assumes that all failures belong to one or of the seven categories. The seven categories are:

- Faulty design
- Material defects
- Processing or manufacturing errors
- Assembly or installation defects
- Off-design or unintended service conditions
- Maintenance deficiencies
- Improper operation

A straightforward approach to identifying root causes can be introduced at Lely by using these root cause categories in the service data. These root cause categories can also be used as a starting point for service technicians' uniformly interpretable root cause categories.

Machine error codes can next to error causes contribute to the determination of breakdown causes. To check whether an error is related to a breakdown, it is also important to store the error moment. The Collector saves all critical error codes including the error moment to its internal memory. A service engineer can read the Collector's last error code and enter this information in the service report. This action can also be performed automatically when a Collector is connected to the cloud via IoT. It is additionally possible to analyze all Collector error codes, instead of only the last one, when this information is saved to the cloud. Studying error codes is also a method to predict component failures (Wang et al., 2017) (Gutschi et al., 2019).

Both root cause categories and error codes are recommended to save as maintenance data. The maintenance data should contain background information about the cause of service visits. For example, information about the type and reason for a visit, the machine serial number, the duration of a visit and part usage. The maintenance (as-maintained) data can also be combined with production (as-build) data in one database to quickly trace all activities at a machine. According to Kans and Ingwald (2008), such a common database tries to describe factors that could influence the life cycle. Kans and Ingwald (2008) database model contains relevant data to support maintenance on various aspects. However, this database model does not include the service visit-centered view on maintenance systems. For required maintenance information during service visits, we look

at the paper of Okogbaa et al. (1992) and the paper of Kallen and Noortwijk (2005) for periodical condition data.

A combination of Kans and Ingwalds database model, the service visits information from Okogbaa et al. and periodic condition data of Kallen and Noortwijk should lead to balanced advice about what service data to collect. We start with the maintenance information that should be collected according to Kans and Ingwald (2008). Information about the type of breakdown, the downtime and the cause should be saved during machine breakdowns. The costs, planned and the actual time during maintenance are other relevant information presented in the database model.

According to Okogbaa et al. (1992), additional recommended information during service visits is work orders, equipment list, spare part usage, date of visit, service engineer, down type and time based information like date down and up time. The equipment list should contain information about the installation, configuration, code and the systems equipment name. It is also relevant to store the type of maintenance which Okogbaa et al. refer to as down type. The kind of maintenance visit is used to classify the reason for the machine downtime by Okogbaa et al. (1992).

Lastly, data about the condition of systems and components. This data can be collected by periodical inspections or with condition monitoring. Maintenance visits can better be scheduled when real-time condition data is available from condition monitoring. Condition data gathered from periodical inspections is another solution when condition monitoring is not an option. The periodical condition information is valuable information for a deterioration process that can be collected during maintenance visits. This condition information can be either a condition measurement or a condition/damage ranking (Kallen and Noortwijk, 2005).

Advice for Lely

The knowledge from the literature about maintenance service data is applied to the situation at Lely in this section. First of all, the different types of maintenance visits should be used uniformly and correctly. This makes the information collected from a visit more useful. Additionally, general information of a service visit, such as date, duration, part usage and machine visited, should be collected for every visit. This information is already well stored in Lely's current service database. Furthermore, it is desirable to store additional information during certain visit types. This research focuses on maintenance and therefore, we zoom in on information collection of maintenance-related visits such as breakdowns, repairs and preventive maintenance visits. Table 3.1 provides an overview of the recommended to collect during a maintenance-related visit type.

A breakdown visit is defined as a maintenance visit when a machine stops working. It is desired to collect additional information about the error description and visit results during a breakdown. Furthermore, the error cause or the underlying reason why a machine (is) stopped working is useful to collect. The use of categories for collecting information makes it less time-consuming to collect and easier to analyze. The recommended categories options for the error symptoms, error causes and visit results are listed in Table 3.2 and are clarified later in this section. The system's last error code and moment provide more detailed information about the underlying cause of a breakdown. The machine downtime during a breakdown is further relevant information to collect for machine performances. Most of this information is also valuable to collect during repair visits.

Repair visits are scheduled visits to repair or replace parts at a machine. The difference between a repair and a breakdown is the status of a machine and the schedulability. The machine is not working anymore during a breakdown, whereas the machine still works (with less performance) during a repair. Information about the last error code and downtime of a machine is therefore not necessary to collect at scheduled repairs. Besides that, a visit during a breakdown is not schedulable, while a repair visit is schedulable.

Table 3.1: Advice for data collection

<table>
<tr><th rowspan="2"></th><th rowspan="2">General information</th><th colspan="3">Policy type</th></tr>
<tr><th>FBM</th><th>AMB / UBM</th><th>CBM</th></tr>
<tr><td rowspan="2">Breakdown</td><td rowspan="4">Service-order number
Visit type
Customer
Customer
Part usage
Visit duration
Service engineer
Date of visit
Time to failure
Serial machine number</td><td colspan="3">Error description category
Comment error description
Error cause category
Latest machine error code (IoT)
Date latest machine error code (IoT)
Visit result category
Machine downtime</td></tr>
<tr><td></td><td>Age / usage (IoT) replaced part</td><td>Condition data replaced part</td></tr>
<tr><td rowspan="2">Repair</td><td colspan="3">Visit trigger category
Error description category
Comment error description
Error cause category
Visit result category</td></tr>
<tr><td></td><td>Age / usage (IoT) replaced part</td><td>Condition data replaced part</td></tr>
<tr><td>Preventive maintenance</td><td></td><td>Age / usage (IoT) inspected / replaced part</td><td>Condition data inspected part</td></tr>
</table>

The recommended options for the error symptoms, error causes and visit result categories are clarified in this paragraph. Table 3.2 provides an overview of all recommended options for each category. The *"Visit trigger category"* consists of five generic categories and a general option which are found together with an expert at Lely. It can occur that the five categories do not cover all the underlying reasons why a machine stopped. Therefore, the general option is added. A free text field should provide more information about the maintenance visit trigger when selecting the option "other". The error description category categorizes the type of error during a breakdown or repair, such as the failure mode category of Okogbaa et al. (1992). Generic error description categories could be damage or leakage. Less generic categories will become more machine-specific and result in a larger number of options. Examples of general error descriptions for the Collector are shown in Table 3.2.

Table 3.2: Advised data categories

Visit trigger category
Animal health
Related to technical alarms
Output does not meet expectations
Strange behaviour equipment
Detected during service visit
Other

Error description category
Cleaning issue
Driving issue
Connection issue
Signal issue
Vacuum issue
Electrical issue

Error cause category
Material defects
Improper operation / external influences
Software issues
Repeat visit
Maintenance deficiencies
Installation / assembly defect

Visit result category
Solved: Part replaced
Solved: Part repaired or cleaned
Solved: Reset / update software
Follow-up visit required
No solution

The seven error cause categories from Geitner and Bloch (2012) are used as the starting point for unambiguous and straightforward error cause categories. It is for a service engineer hard to determine whether a part is incorrectly designed. Therefore the options *"Faulty design"* and *"Off-design"* options are not included. The category *"Processing or manufacturing errors"* is changed to *"Software issues"* to make it unambiguous for service engineers and more suitable for Lely. The category *"Repeat visit"* is included after discussions with employees of Lely. Finally, there are five possibilities for the visit result category. This category adds information to the as-maintained data of a specific machine and provides an overview of the maintenance visit results. The options in this category are aligned with an expert of Lely and indicate the result of a visit and what was performed to solve the problem.

Additional recommended maintenance data to collect depends on the maintenance policy of a component. Measuring the condition of a component is only meaningful if the degradation of a component is measurable via the condition. It is even possible to predict component failures by using the current component condition and historical component data. Where condition data is required for condition-based maintenance, are age and usage data useful information for calendar time-based maintenance and usage-severity based maintenance, respectively. This data is relevant to collect during the different visit types. It is not necessary to measure the condition of all components during every visit type. Condition measurements are required for CBM components during preventive maintenance visits and replaced CBM components during breakdowns and repairs. The usage and age information is necessary to replace USBM and CTBM components and inspected parts during preventive maintenance visits.

3.3 Data processing

The data processing of Lely's service data is performed in this section. Data processing consists of two sub-steps: data cleaning and data preparation. Data cleaning is the first step in data processing, which is followed by data preparation. This section is divided into two subsections that describe the two steps of data processing of the available service data at Lely. The cleaned and processed data is used as input for analysis later in this thesis.

3.3.1 Data cleaning

Data cleaning is the first step in data processing. This step aims to create a clean, error-free data set that is used for analysis. Data cleaning is essential at manually entered data because this data is likely to contain errors. The data cleaning method and steps differ based on the available data. Data processing is mostly performed before data analysis. Without performing data cleaning, the data analysis is unlikely to provide the correct results due to the so-called "garbage in garbage out" concept. The data cleaning process is described in this section.

The data cleaning process described in this section results in a clean data set for the analysis. Figure 3.2 provides a schematic representation of the data cleaning process. The cleaning process starts with the complete service database from 2016 until the beginning of July 2021. The service data set consists of more than 40.000 lines with information about almost 13.000 maintenance actions of more than 1.500 individual Collectors. Data cleaning is performed to obtain the only complete maintenance data from individual Collectors.

The data cleaning step results in a data set with machines numbers for which maintenance data is appropriately logged. This step is necessary because the Lely Centres fill in the data differently in this database. Some Lely Centres only register parts for which they claim warranty in this database, while others report all maintenance actions. To filter out the data of the Lely Centers that only enter parts for warranty in the system, the cleaning rule of at least one PM visit of a machine is implemented. With this cleaning rule we assume that when one or more PM activities are registered for an individual Collector, the maintenance activities are properly documented.

Data cleaning step:

- All machine numbers without preventive maintenance activities are filtered out of the data. There is assumed that when at least one preventive maintenance activity is found for a machine in the service database, the maintenance data for this machine is logged correctly.

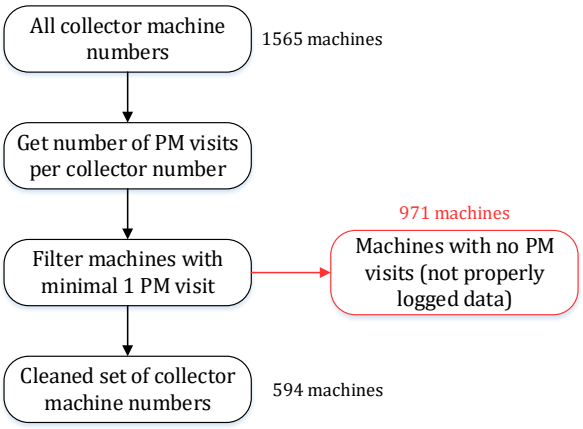

Figure 3.2: Data cleaning steps

As shown in Figure 3.2, the data cleaning step results in a data set with 594 individual machines. There are 6013 maintenance actions performed on these machines, which result in 19.335 lines of maintenance data.

3.3.2 Data preparation

The machine numbers with properly logged service data are used as starting point for the data preparation. Before the data can be used, it must be prepared to contain the right information in the correct form. The lifetimes of failed and non-failed components are required as input to determine the failure behaviour of components. Right-hand suspended or right-censored data are commonly used terms in literature for non failed data. Since Lely's produced machines have different ages and PM performed for specific components, we are dealing with multiply censored data. This should be taken into account when using this data. Figure 3.3 shows a schematic overview of the performed data preparation steps.

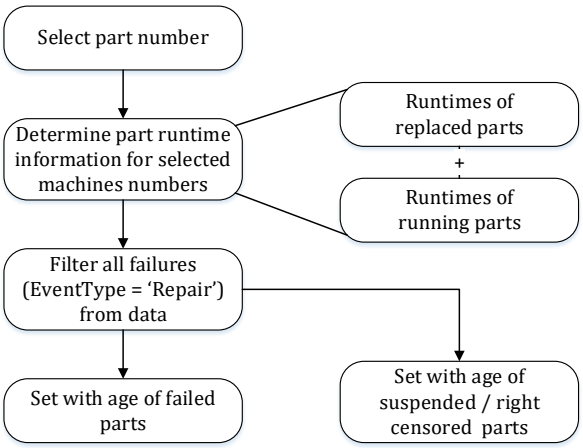

Figure 3.3: Data preparation steps

Data preparation steps:

1. The first processing step is selecting a component number for which the data is prepared.
2. The second step is to determine the components' runtimes of the machines with properly logged maintenance data. This step is divided into two smaller sub-steps:
   - **Runtimes of replaced parts.** The runtime of the replaced parts is calculated with the date of replacement minus the installation date of a machine. When a component is replaced more than once at the same machine, the runtime is determined with two replacement dates.
   - **Runtimes of running parts.** The difference between a given date and the machine installation date is used to compute the part runtimes when this part has not yet been replaced. The runtime of newly installed components that are still in a machine is determined by the difference between a specified date and the date the new component was installed in the machine.
3. The last step is creating data sets with ages of failed and suspended parts. It is assumed that all parts replaced with the event-type "*Repair*" are failures. All parts with event-type "*Repair*" are filtered out to get a set with failures. The data without event-type "*Repair*" are the suspended parts.

Chapter 4

Literature review

The literature study conducted in this chapter will contribute to addressing the following research question:

2. *Which maintenance optimization model is best suited to develop a maintenance concept for Lely's machines, and which planning horizon assumptions best fit Lely's features and requirements?*

This research question is answered in three sub-sections. First, the different time horizons are explained in section 4.1. The knowledge about the time horizons and the model requirements of section 2.5 are used as the starting point of the literature review in section 4.2. The conclusion of the literature review is given in section 4.3, where the models used as a starting point for this study are presented.

4.1 Time horizon

The planning horizon in a maintenance model is the length of time considered in the future incorporated in a particular plan. Nicolai and Dekker (2008) classify the planning horizon into three types: infinite, finite and rolling. This section is divided into three subsections that clarify the planning horizons and present the properties of each horizon.

Infinite horizon

The majority of articles in this maintenance modelling field assume an infinite planning horizon. These models are called stationary since they provide maintenance rules which do not change over the planning horizon (Nicolai and Dekker, 2008). This implies that the control parameters, such as the age limits, control thresholds and the maintenance interval τ, will always be the same (stationary). Models with an infinite horizon will obtain the same stationary solutions independent of the starting moment in time and component state at the beginning of the model. A maintenance model with an infinite horizon assumes a static solution for the long-term maintenance frequencies.

Stationary models use an infinite horizon and assume a long-term stable system solution (Dekker and Wildeman, 1997). This infinite horizon can be assumed when the maintenance interval is small (days or weeks) compared to a life cycle (10 a 20 years) of a system (Zhu, 2015). A system solution with a static maintenance interval is more convenient to implement in practice for planning reasons. A stable solution can be furthermore suitable for spare part inventory management. A disadvantage of the infinite horizon is that newly available information gathered during the lifetime of a machine cannot be implemented in the same model. New information can be the varying use of components in a system or changes in failure behaviour, making another control limit better than the stationary control limit. An option to use new information in an infinite horizon model

is by changing the initial input parameters of the model and solve it again. However, this is a model run with other input parameters instead of implementing the new data in the infinite model. Therefore, implementing new information in a model with an infinite horizon is not possible. As a result, an infinite horizon model might miss important new available information that could prevent machine breakdowns.

Finite horizon

Models with a finite horizon are called non-stationary. Their maintenance rules may change during the time horizon. Finite horizon models can use newly available information such as varying component degradation or unexpected maintenance opportunities (Nicolai and Dekker, 2008). These finite models only consider systems in a predetermined time horizon. They therefore assume that the systems are not used afterward unless a residual technical lifespan is incorporated at the end of the horizon (Dekker and Wildeman, 1997). The starting state of the system and the predetermined time horizon influence the model solution. Additionally, most models with a finite horizon use a dynamic grouping of components which mostly results in dynamic maintenance intervals. Maintaining components in varied group compositions (dynamic grouping) leads mostly to varying (dynamic) optimal maintenance intervals between groups. The solutions of finite models are therefore mostly dynamic and non-stationary, which allows changes in the maintenance interval, age and control limits over time.

The advantage of the finite planning horizon is the incorporation of newly available information. The use of this new information may create opportunities for performing maintenance at lower costs. The new information could also lead to changeable, unstable solutions. A challenge for multi-component models with a finite time horizon is making the dynamic, unstable maintenance scheduling operational.

Rolling horizon

A model with a rolling time horizon uses a long-term (i.e., infinite horizon) plan adapted with newly available information. This new information is included in the model when it becomes available or after performing a maintenance action. The usage of new information in a modifiable finite horizon is a major difference between the rolling and finite horizon. The use of new information can change maintenance rules, such as control limit, in the planning horizon, making the rolling horizon a non-stationary model (Wildeman et al., 1997).
The rolling horizon model bases the short-term system decisions (finite horizon) on the long-term component plans (infinite horizon). In this way, models with a rolling horizon try to close the gap between models with finite and infinite horizons and benefit from both planning horizons advantages (Dekker and Wildeman, 1997). The use of new information in combination with stable long-term plans should result in a model with more stable solutions than the finite horizon models and a less stable solution than the infinite horizon. The dynamic grouping of components from the finite horizon also results in dynamic maintenance intervals in most rolling horizon models.

An advantage of a rolling horizon model is using new information combined with long-term (infinite) plans. This should result in an adaptable model with a more stable solution than models with a finite horizon. However, the dynamic grouping of components is also a challenge in making the maintenance scheduling operational for the rolling horizon.

Table 4.1 provides an overview of the characteristics of the different time horizons in the maintenance models. Lely's model requirements from section 2.5 and the properties of the planning horizon in Table 4.1 show that no horizon perfectly matches their requirements. A model with an infinite horizon meets the static maintenance interval requirement. However, changes in maintenance rules caused by new information are not possible on this horizon. The use of new information is only possible in models with a finite or rolling horizon. Nevertheless, maintenance models with

Table 4.1: Model Horizon

Model horizon	Model solution	Maintenance interval (τ)	Use of new information	Stationary / non-stationary
Infinite	Stable	Static	No	Stationary
Finite	Unstable	Mostly Dynamic	Yes	Non-stationary
Rolling	Combination of stable and unstable	Dynamic	Yes	Non-stationary

a finite or rolling horizon usually have a dynamic maintenance interval. A literature review is conducted in the following section to find a model that closely matches Lely's model requirements. All three modelling horizons are considered. However, the finite and rolling horizon are preferred due to the update possibility with new information. With the update possibility, it is also possible to model replacements of components with a new updated version which sometimes occurs at Lely. Modificative maintenance is performed when a component is not working properly and therefore is replaces with a technically more advanced component. This is not possible to include in a model with an infinite horizon because of the static policies. Therefore, a rolling or finite horizon in a model is preferred. Additionally, a finite horizon might be more appropriate when the expected lifetime of one or more components is close to the intended system life.

4.2 Literature study

The literature reviews of Nicolai and Dekker (2008) and de Jonge and Scarf (2019) are the starting point of this literature study. Both reviews provide an overview of the researched areas in maintenance models, such as the dependencies between components, deterioration processes and planning horizons. These literature reviews distinguish between single and multi-component maintenance models. The literature review conducted in this thesis is focused on the maintenance optimization of multi-component systems with different planning horizons. However, the number of models that combine CBM, UBM and FBM components in the various planning horizons is limited. Requirements related to Lely's specified model are discussed in section 2.5. Table 4.2 provides an overview of the model characteristics discussed in this section.

Table 4.2: Maintenance models

Author	Number of components	Individual policies	Minimal repair	Maintenance interval	Failure interaction	PM opportunities	Updateable	Planning horizon
Koochaki et al. (2012)	Example with three	UBM & CBM	No	Dynamic	Positive economic	SD & USD	Condition update	Finite (10 years)
Pandey et al. (2016)	Example with fourteen	UBM	Yes, only at CM	Static	Positive economic	SD	Update age	Finite
Wu et al. (2020)	Example with eight	UBM & CBM	No	Dynamic	Positive economic	SD & USD	Update component information and scheduling	Rolling
Vu et al. (2014)	Example with sixteen	UBM	Yes, only at CM	Dynamic	Positive and negative economic	SD	Update component information and scheduling	Rolling
Shi et al. (2020)	Example with ten	CBM	No	Dynamic	Positive economic	SD	Bayesian parameter updating and system reliability updating	Rolling
Zhu (2015)	Example with fifty	FBM, UBM & CBM	No	Static	Positive economic	SD & USD	No updating	Infinite
Hoedemakers (2020)	Example with nineteen	FBM, UBM & CBM	No	Static	Positive economic	SD only	No updating	Infinite
Laggoune et al. (2009)	Example with two	UBM	No	Static	Positive economic	SD & USD	No updating	Infinite
Xu et al. (2021)	Example with two	CBM	Yes	Static	Stochastic and positive economic	USD	No updating	Infinite
Zhao et al. (2019)	Example with four	CBM	Yes	Dynamic	Positive economic	USD	Update condition in simulation	Rolling
Liu et al. (2020)	Example with fourteen	UBM	Yes	Dynamic	Positive economic	SD	Update age	Finite

Koochaki et al. (2012) created a small serial system simulation model with three components. The maintenance policies of these components are ABM and CBM or a combination of both. Failures of the components occur according to a gamma distribution in a simulation with discrete events. The component conditions of CBM components are continuously monitored and used in the simulation. This model aims to minimize maintenance costs and maximize the system availability using opportunistic maintenance during SD and USD. A discrete simulation model was performed for a finite period of 10 years.

Liu et al. (2020) developed a maintenance model for a multi-states system in a finite horizon. This model maximizes the number of missions/tasks that can be successfully performed. At the end of the last mission, the effective component age is known. The effective component age can be seen as the working hours of a component. Based on this information, the model decides whether to replace or repair components to maximize the number of missions in a finite horizon. Liu et al. use a neural network to find the model solution. An example of a system with fourteen components demonstrates the model.

Another preventive maintenance scheduling model with a finite horizon is proposed by Pandey et al. (2016). This model includes limited resources and finds the number of maintenance breaks with minimal costs is a finite horizon. The model of Pandey et al. uses a minimal reliability level to determine whether preventive maintenance in required. This model results in static maintenance intervals determined by dividing the finite horizon by the number of maintenance breaks.

Wu et al. (2020) developed a dynamic multi-component maintenance model with a rolling time horizon with ABM and CBM component policies. The rolling time horizon in this model ensures optimal scheduling of only the next maintenance visit. The next optimal preventive maintenance visit scheduling is based on optimal age and conditions for preventive replacements with actual component information after every maintenance visit (corrective and preventive). A numerical example of a system with eight components that followed a Weibull lifetime distribution tested this model. However, this model should also be able to handle large systems, according to the authors.

Vu et al. (2014) presented a maintenance model with a dynamic grouping strategy. This model determines the optimal maintenance moments for an individual component over a long period (infinite horizon). These moments are used for finding the optimal grouped maintenance moments on the system level. A genetic algorithm is used for finding the optimal grouped maintenance moments. These moments can be updated as soon as new short-term information of components becomes available. The first step is to find optimal maintenance moments at the component level using the new information. This is repeated, after which new maintenance grouping moments are determined.

Shi et al. (2020) developed a condition-based system reliability decision-framework for multi-component systems. The component failures are assumed to be hidden in this framework and can only be detected during inspections. Shi et al. (2020) use predictions of component readabilities to compute the reliability of the sub-systems and the system. A Bayesian updating method is used to improve the estimation of the component degradation parameters.

Another multi-component maintenance model for condition-based components is presented by Zhao et al. (2019). A Weibull proportional hazard model (WPHM) based on the past failure and vibration monitoring data is used to model the deterioration of component conditions. The condition of the modelled components is monitored. A component is replaced when it exceeds the threshold for condition-based maintenance. At this moment, opportunistic maintenance (OM) is performed at components which condition exceeds the OM condition threshold. The model of Zhao et al. contains both stochastic and positive economic dependencies between the components.

Zhu (2015) proposes a multi-component maintenance model that uses a mixture of component policies (FBM, ABM and CBM) for a long period (infinite horizon). This model allows opportunistic maintenance during Scheduled down (SD) and Unscheduled down (USD). A heuristic is used to optimize the system maintenance interval and the thresholds of ABM and CBM components. No specific degradation models or lifetime distributions are assumed in the model of Zhu. This model is verified with a numerical example of a system with 50 components. Zhu also recommends including Bayesian updating for degradation paths which should provide more dynamic maintenance solutions.

Laggoune et al. (2009) present a multi-component maintenance model with UBM components. This model only considers perfect replacements of failed or preventive replaced components. Preventive replacements of components can happen during both SD and USD. The Weibull distribution is used to model the failures of the two components in an example of the simulation model.

Xu et al. (2021) investigated the optimal policy for a multi-component K-out-of-N system with only CBM components. Xu et al. developed a model for a system with periodic conditions inspections and imperfect maintenance activities. Stochastic and positive economic dependencies are considered in this model. A Monte Carlo simulation is used to find the optimal inspection interval in a numerical example of a 1-out-of-2 system.

Hoedemakers (2020) used a multi-objective maintenance model for Lely for a long period with static maintenance intervals. Hoedemakers' maintenance model combines the cost objective of Zhu (2015) and the downtime objective of Peng and Zhu (2017) in one model. This model allows only opportunistic maintenance during SD, which has Lely's preference. A case study with seventeen components and a mixture of different component policies tested Hoedemakers (2020) model. This model does not include the updating of failure parameters or component conditions.

4.3 Conclusion

Based on the literature review, the models of Zhu (2015) and Shi et al. (2020) are selected as starting point for this research. The model of Zhu is chosen based on the positive economic dependency, the different individual maintenance policies and the static maintenance intervals. The model of Shi et al. (2020) is selected because of the planning horizon, the updating method of lifetime parameters and the use of perfect repairs. In previous research, Hoedemakers combined the models of Zhu (2015) and Peng and Zhu (2017) for his two objective functions.

Combining both models and the multi-objective approach provides a multi-objective model with different individual maintenance policies with the updating of lifetime parameters. The selected model of Zhu has an infinite time horizon. However, a finite or rolling horizon is preferred because of the replacements of components with new updated versions of a component. A finite horizon might be more appropriate when the expected lifetime of one or more components is close to the intended system lifetime. Therefore, the planning horizon in the model of Zhu (2015) should be adapted. Changing the time horizon of Zhu's model requires a change in the solution approach as well. The decomposition of a system from Zhu is implemented to find a solution through simulations with a finite horizon. Shi et al. lifetime parameter updating method is applied at a component level and implemented in the simulation horizon.

A multi-objective multi-component maintenance model with parameter updating capabilities in a finite horizon is created. As far as we know, such a maintenance model is not available in the literature yet. The updating of component lifetime parameters should result in a more realistic and more reliable lifetime distribution.

Chapter 5

Multi-component maintenance model

This Chapter presents the multi-component maintenance model that is used in this research. A multi-component maintenance model with parameter updating capabilities in a finite horizon is presented. Section 5.1 describes the modelling at the system level with the use of decomposition. Sections 5.2, 5.3 and 5.4 explain the modelling at the component level with different maintenance policies.

5.1 System description

The system of Lely that is modelled in this chapter is a multi-component system where components are assumed to be connected in series. When one component fails in a serial system, the complete system shuts down. A broken component needs to be repaired or replaced before the system can function again.

In this model, a serial system is decomposed into component levels. This model considers both Scheduled down (SD) and Unscheduled down (USD). A SD is a planned moment when maintenance is performed on the system. This SD is performed with a maintenance interval τ. At these moments, preventive maintenance can be performed for ABM and CBM components. The setup costs and downtime of an USD can be saved when components are maintained during a SD. A CM action can be prevented by maintaining a component during a SD. By avoiding a CM, a service engineer does not have to unexpectedly visit a farm for maintenance, which saves the setup costs and downtime of a USD.

An USD is an unplanned system down that happens when a component fails between two SD. A CM action is performed for the failed component during an USD. The failures of FBM components always result in USD. ABM and CBM components can also result in an USD when they fail before a preventive replacement. In this research, only CM is considered on the failed component during USD. We do not use the USDs as an opportunity for PM for other components.

5.1.1 Assumptions

The following assumptions are made in the maintenance model that is described in this chapter.

1. Degradation of CBM components is independent of other components in the system and their SD and USD
2. The lifetime of ABM components is independent of SD and USD caused by other system components

3. The time horizon is finite
4. Maintenance actions restore components to an as good as new state
5. The system is composed of components that are independent of each other
6. Maintenance actions occur at fixed maintenance moments $n\tau, n \in \mathbb{N}$
7. The considered system is a serial system

5.1.2 Model description

In this model, we aim to optimize the maintenance concept of our system. This implies an optimal maintenance interval for the system and optimal maintenance thresholds at the component level. The maintenance period is a decision variable at system level, whereas the maintenance thresholds are the decision variables at a component level. The maintenance threshold for the CBM components is the control limit C_i and for ABM components, the age limit A_i. Solving both the component maintenance thresholds and maintenance interval τ simultaneously results in a significant increase of the solution space. Therefore the decomposition approach used in the work of Zhu (2015) is applied to solve this model. The problem is decomposed into system level and component level.

In the decomposition approach, the maintenance threshold for ABM and CBM components are first optimized at the component level for a given τ. After the component optimization, the maintenance interval is optimized at a system level in an iterative procedure. In this decomposition approach, Zhu (2015) used a mixture of single-component models. Different maintenance policies are considered at the component level: FBM, ABM and CBM. The maintenance interval for a SD is fixed at the component level, and the component thresholds are the decision variables. This results in two independent problems that can be solved in a specific order by enumerating various maintenance interval values. The system solution is found by combining all decomposed component solutions to the system level. At a system level, a preferred system solution is selected. The decision variable at the system level is the maintenance interval τ.

The system lifetime is assumed to be finite and therefore, the expected lifetime of the system is used as finite time horizon T in the model. The system's expected lifetime was estimated by experts of Lely. The system, components and maintenance process are simulated over a finite horizon [0, T]. Different component models used for simulation are explained in the following sections. The component models also include several failure parameter updating moments during the time horizon [0, T] denoted by $t_1, t_2, ..., t_n$ where $t_n < T$. At these moments, the failure parameters are updated based on newly available service data. The component models will likely give a better impression of the real situation by including an updating mechanism with new data.

Let us denote with I the set of all numbered components in a system from 1 to k; $I = \{1, 2, ..., k\}$. Different maintenance policies are applied to the component in this system. Subset I_{CBM} is used to identify components that follow a condition-based maintenance policy. With subsets I_{ABM} and I_{FBM} we denote the components under age-based and failure-based components, respectively. All components in the system together result in the set of all components $I = I_{CBM} \cup I_{ABM} \cup I_{FBM}$ and every component belong to only one set, $\emptyset = I_{CBM} \cap I_{ABM} \cap I_{FBM}$.

The average yearly system costs are given by the sum of all average yearly component costs per year Z_{T_i} in time horizon T plus the setup costs for a SD. This results in Equation 5.1. The average yearly component costs are determined by solving the single-component maintenance model, explained in the following sections.

$$Z_{Tsys}(\tau) = \frac{\left\lfloor \frac{T}{\tau} \right\rfloor * c^{SD}}{T} + \sum_{i \in I_{FBM}} Z_{T_i} + \sum_{i \in I_{ABM}} Z_{T_i}\left(A_i(\tau)\right) + \sum_{i \in I_{CBM}} Z_{T_i}\left(C_i(\tau)\right) \tag{5.1}$$

In this equation $\frac{\lfloor \frac{T}{\tau} \rfloor * c^{SD}}{T}$ is the average yearly setup costs of scheduled downs and Z_{T_i} is the average yearly costs of component i in time horizon T. $A_i(\tau)$ and $C_i(\tau)$ denote the age and control limit of component i with a maintenance interval of τ. These age and control limits are obtained from the of the single-component models and are dependent on τ. At a system level, the main goal is to determine the SD interval that minimizes the yearly average costs:

$$\min_{\tau} \; Z_{T_{syst}}(\tau)$$

$$\textit{subject to} \quad \tau^{LB} < \tau < \tau^{UB}$$

We use a set of maintenance intervals τ with lower bound τ^{LB} and upper bound τ^{UB} as input from the decision-maker. A lower bound is specified because systems users may not allow too many scheduled down or due to limitations in the capacity of service engineers. Regulations specifying the minimum frequency of inspections can give the value of the upper bound.

An iterative technique is used to solve the optimization problem at a system level. For a maintenance interval τ, the average yearly component costs with maintenance threshold ($A_i(\tau)$ and $C_i(\tau)$) are determined for each component. The component thresholds with minimal average costs are used to calculate the minimal average system costs $Z_{T_{syst}}(\tau)$ using equation 5.1. This procedure is repeated for all possible values of τ between τ^{LB} and τ^{UB} to find the maintenance interval with the minimum average yearly system costs $Z_{T_{syst}}$.

The same procedure for finding the minimum system costs is used to find the minimum system downtime $V_{T_{syst}}$. The system downtime is determined with the following equation:

$$V_{Tsys}(\tau) = \frac{\lfloor \frac{T}{\tau} \rfloor * d^{SD}}{T} + \sum_{i \in I_{FBM}} V_{T_i} + \sum_{i \in I_{ABM}} V_{T_i}\left(A_i(\tau)\right) + \sum_{i \in I_{CBM}} V_{T_i}\left(C_i(\tau)\right) \tag{5.2}$$

where $\frac{\lfloor \frac{T}{\tau} \rfloor * d^{SD}}{T}$ is the average yearly setup downtime of scheduled downs. The average yearly downtime of component i in time horizon T is given by V_{T_i} in equation 5.2. The main goal is to determine the SD interval that minimizes the yearly average downtime at a system level.

$$\min_{\tau} \; V_{T_{syst}}(\tau)$$

$$\textit{subject to} \quad \tau^{LB} < \tau < \tau^{UB}$$

5.1.3 Simulation approach

In this research, a simulation is used to find the system solutions. Simulation is an effective approach to approximate difficult problems. Especially when solving a problem exactly or approximately is too complex or too time-consuming. In our study, we have to deal with a multi-objective multi-component maintenance model with a finite horizon. This model is too complex to solve exact, therefore we use simulations. A simulation approach is a commonly used method to solve complex problems. Simulation results approach the exact solution when the number of simulation runs increases (Boon et al., 2020).

In the following sections, the single-component models are described. The component threshold is considered as a decision variable and the maintenance interval is a fixed input value in these models. Simulations solve the single-component models to find the average component costs and downtime a certain maintenance threshold and maintenance interval. Multiple simulations with different maintenance thresholds and intervals will be performed to find the results of a single-component model. For CBM components, random degradation is simulated in the single-component models to model the degradation. For ABM and FBM components, random failure

moments are simulated to model the failures of these components. To reduce the variability of simulations, realizations of random degradation and failure moments will be used in component simulations in all scenarios with different decision variables. When a component has four options as maintenance thresholds and the system considers five different values of τ, the same realizations of degradation or failure moments is used in all these 20 simulations scenarios. The realization of the component degradation or failure moments used as input in these 20 simulations is named a set. For each component in the system, a set of degradation or failure moment is realized. The combination of sets for all different components $i \in I$ is called a trace. The number of traces used in the simulation model indicates the number of simulations that have been performed.

The total costs and total downtime incurred during the maintenance performance in a simulation are tracked for all simulations of a component. These costs and downtime are converted to average annual costs and average annual downtime by dividing the total costs or downtime by time horizon T. The average of all simulation traces for a component with threshold C_i or A_i is used to calculate the final average yearly costs and downtime. This results in solutions for every maintenance threshold in each maintenance interval for all components. The following formulas are used to find the final average cost results from the simulation where T_r denotes the number of traces:

$$Z_{T_i}\left(C_i(\tau)\right) = \sum_{n=1}^{T_r} Z_{T_{i,n}}\left(C_i(\tau)\right) \ / \ T_r \qquad \forall i \in I_{CBM} \tag{5.3}$$

$$Z_{T_i}\left(A_i(\tau)\right) = \sum_{n=1}^{T_r} Z_{T_{i,n}}\left(A_i(\tau)\right) \ / \ T_r \qquad \forall i \in I_{ABM} \tag{5.4}$$

$$Z_{T_i}(\tau) = \sum_{n=1}^{T_r} Z_{T_{i,n}}(\tau)) \ / \ T_r \qquad \forall i \in I_{FBM} \tag{5.5}$$

The final average downtime results from the simulation are found with the use of the following formulas:

$$V_{T_i}\left(C_i(\tau)\right) = \sum_{n=1}^{T_r} V_{T_{i,n}}\left(C_i(\tau)\right) \ / \ T_r \qquad \forall i \in I_{CBM} \tag{5.6}$$

$$V_{T_i}\left(A_i(\tau)\right) = \sum_{n=1}^{T_r} V_{T_{i,n}}\left(A_i(\tau)\right) \ / \ T_r \qquad \forall i \in I_{ABM} \tag{5.7}$$

$$V_{T_i}(\tau) = \sum_{n=1}^{T_r} V_{T_{i,n}}(\tau)) \ / \ T_r \qquad \forall i \in I_{FBM} \tag{5.8}$$

Sufficient independent simulation traces are required to obtain average results without outliers caused by randomness in the simulation. A higher number of simulation traces should result in a more accurate simulation solution. More simulation traces also results in a higher computation time. The number of simulation traces can be determined by running numerous simulations with the same amount of traces and considering the volatility of the numerous simulation results. The standard deviation of the results presents the volatility of the results. When the standard deviation of the results is acceptable, a sufficient number of simulation traces is found.

In the following sections, the single-component maintenance models are explained. The simulation approach is adopted to evaluate the cost and downtime for each maintenance policy. First single-component maintenance model of CBM components is explained, followed by the models of ABM and FBM components in the following sections.

5.2 Condition-based components

A condition-based maintenance policy measures condition of a component and shows the component state. There are two commonly used thresholds in condition-based maintenance: the condition/warning threshold and the failure threshold. A CBM component can be modelled with three states in a semi-Markov model. The condition and failure threshold can be represented where a component moves through the three degradation states sequentially. The semi-Markov model consists of three states and is shown in Figure 5.1.

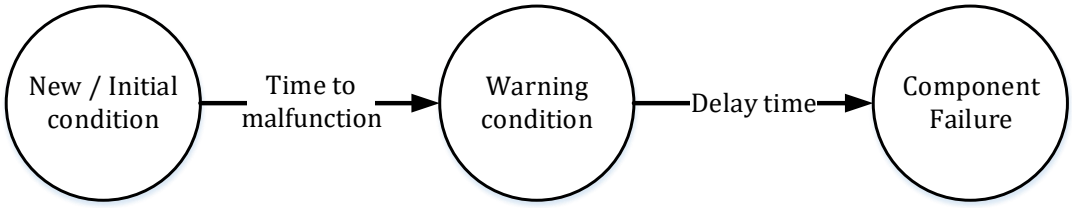

Figure 5.1: Delay time model as a semi-Markov model (adapted from Arts (2017))

The first state is the initial/new condition of a component. After some time, named the time to malfunction, the process enters the condition warning state. The final failure state is reached after some time, named the delay time. A delay time model is a specific instance of a Markovian degradation process. It implies that a component's failure occurs in two stages: it first appears to be defective and it fails after a certain amount of time (Baker and Christer, 1994). A component should be replaced or maintained somewhere between these two thresholds to prevent a failure.

In our search to find a model for CBM components, chapter 3 of the PhD thesis of Zhu (2015) provides a most appropriate match. The model for CBM components is based on the CBM components from Chapter 3 of Zhu (2015), which is further investigated by Peng and Zhu (2017). Figure 5.2 shows the way how CBM components are maintained. The SD maintenance moments are planned at fixed moments and are not rescheduled after a maintenance cycle. A maintenance cycle of a component is the period between two consecutive maintenance actions. This is also illustrated in Figure 5.2. The renewal theory can be used to determine the long-term average yearly costs when assuming that the SD maintenance moments are rescheduled at the end of a maintenance cycle. However, this can not be applied due to the finite horizon and the not rescheduled SD moments. The rescheduling of maintenance moments for all components is hard to apply to a real situation with a system that consists of multiple components. Therefore we use fixed maintenance moments that are not rescheduled in this study.

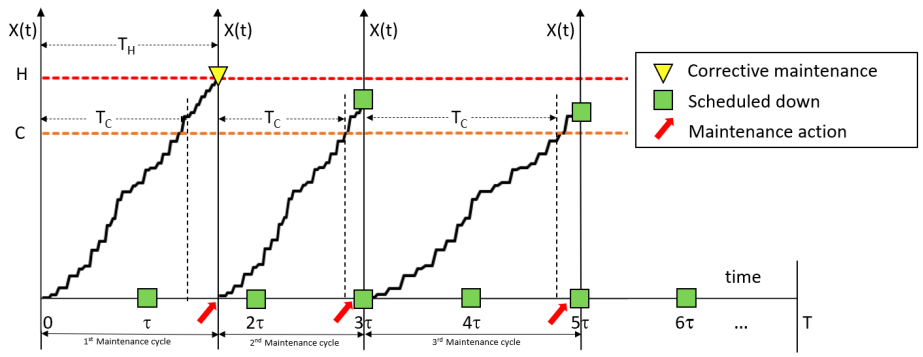

Figure 5.2: Condition-based maintenance policy (adapted from Zhu (2015))

The degradation level of CBM component i is modelled as $X_i(t)$ at time t where $t \geq 0$. The degradation level of a component is assumed to be monitored continuously and to be monotonic increasing. The increase of degradation is randomly realized and stored in sets that are used as input for the simulation model with all scenarios. A component fails in the simulation when the degradation level $X_i(t)$ exceeds the failure level H_i. Because of a serial system assumption, the complete system is assumed to be down when a single component fails. These are also called hard failures. After a failure, a CM has to be performed on the failed component. The cost for a CM action of component i are: $c_i^{CM} = c_i^{REP} + c^{USD}$ where c_i^{REP} is the repair cost of component i and c^{S-USD} is the setup cost for an USD.

A CBM component can also be maintained preventively during a shared SD of the system. Lower preventive maintenance costs (c_i^{PM}) are incurred when PM is performed. By performing PM, the USD setup costs can be saved. Preventive maintenance is performed to a component when the degradation level $X_i(t)$ is above the control/warning threshold C_i and below the failure threshold H_i. The control threshold C_i is the decision variable of component i. This control threshold should the interpreted as follow during a SD:

- If the degradation level $X_i(t)$ is smaller than the control limit C_i, $X_i(t) < C_i$, then do nothing
- If the degradation level $X_i(t)$ is greater than or equal to the control limit C_i and smaller than the failure threshold H_i, $C_i \leq X_i(t) < H_i$, then replace the component i preventively.

The total cost for a CBM component i with maintenance threshold C_i in simulation trace n and with maintenance interval τ is $K_{T_{i,n}}(\tau)$. These costs are all performed CM actions, plus the costs of all performed PM actions. The total downtime ($K_{T_{i,n}}(\tau)$) of a CBM component i with maintenance threshold C_i in simulation trace n and with maintenance interval τ is computed with the same approach for downtime. Both are represented in the following equation:

$$K_{T_{i,n}}(\tau) = b * c_i^{CM} + o * c_i^{PM} \tag{5.9}$$

$$D_{T_{i,n}}(\tau) = b * d_i^{CM} + o * d_i^{PM} \tag{5.10}$$

In these equations, the b stands for the number of component failures and the o for the number of PM actions performed. The average yearly downtime and costs of component i with maintenance threshold C_i in simulation trace n and with maintenance interval τ are determined by dividing the total costs and downtime by the time horizon T in years, which is shown in the equations below.

$$Z_{T_{i,n}}(C_i(\tau)) = K_{T_{i,n}}(\tau) \ / \ T \tag{5.11}$$

$$V_{T_{i,n}}(C_i(\tau)) = D_{T_{i,n}}(\tau) \ / \ T \tag{5.12}$$

5.2.1 Degradation process

There are several methods to model the degradation process of a CBM component. Two commonly used methods in literature and practice are the Random Coefficient Model (RCM) and the Gamma process. The RCM is the most simple process for time-dependent degradation modelling (van Noortwijk, 2009). On the other hand, the Gamma process has proven to be suitable for establishing appropriate inspection and maintenance decisions since they are well suited for modeling temporal variation of degradation (van Noortwijk, 2009). Figure 5.3 shows the degradation of a RCM on the left and the degradation of a Gamma process on the right. The difference between the RCM and Gamma process is also denoted with unit-to-unit variation and time-dependent variation in this figure.

5.2.1.1 Random coefficient model

The linear RCM is an appropriate method for modeling a linear degradation process. This constant degradation rate over time is given by Θ. In the RCM the variation of the model depends on the

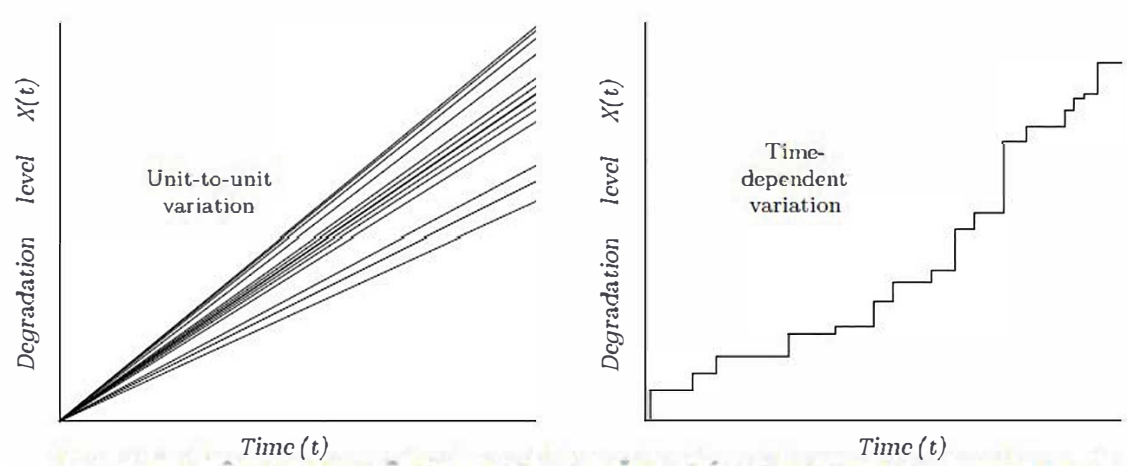

Figure 5.3: Degradation processes of CBM components (from Timmermans (2012))

input value of Θ. This is why the linear RCM is referred to as unit-to-unit variation in Figure 5.3. The degradation in a RCM is modelled with the following equation: $X(t) = x_0 + \Theta t$, where $t \leq 0$ and with x_0 as initial degradation level. A disadvantage of the RCM is the fixed degradation rate. When inspections occur, the modelled degradation could be different than the inspected degradation. This variability in degradation cannot be adapted in the RCM.

5.2.1.2 Gamma process

A Gamma process is a model for degradation processes where it is assumed that the degradation takes place over time with small increments. A Gamma process is suitable for modelling wear, corrosion, degrading health index and other slow degradation that accumulates over time with small increments, according to van Noortwijk (2009). The degradation in a Gamma process is non-decreasing with independent increments over time. The different increments over time are denoted as time-dependent variation in Figure 5.3.
The degradation level at time t_j is given by $X(t_j)$ in a Gamma process. The degradation increment between time t_j and t_{j-1} is the difference between the degradation level $X(t_j)$ and $X(t_{j-1})$. The difference in time between these two moments is given by $\Delta t = t_j - t_{j-1}$. In a Gamma process, the random increments are independent and identically Gamma distributed with $\gamma \Delta t$ as shape parameter and scale parameter η. The degradation increment for component i in time period Δt are i.i.d with a $\Gamma(\gamma_i \Delta t, \eta_i)$ distribution.

In the near future, it will be possible for Lely to monitor the condition data for the Collector by sensors via IoT. With this condition data, more information about a real degradation path can be found. Besides that, the true degradation level of a component can be used in an individual maintenance model. This new information about the degradation behaviour and rate is hard to include in the RCM since it only uses a fixed degradation rate. Changes in the degradation behaviour over time can be incorporated in a Gamma process by adapting the Gamma process parameters since the increments are independently identically distributed. Therefore, the degradation of components in this research is modelled by a Gamma process.

5.3 Age-based components

The simulation model for ABM components explained in this section is based on Chapter 4 of Zhu (2015). We assume that the lifetime of an ABM component i is distributed according to a Weibull distribution with shape parameter β_i and scale parameter α_i. In the simulation model,

random failure moments according to a Weibull distribution are realized in sets that are used as input for the simulation model.

There are two options in performing maintenance for ABM components, namely PM and CM. The ABM component i can be preventively maintained during a SD with associated costs and downtime of c_i^{PM} and d_i^{PM}, respectively. These costs and downtime includes the labour costs and duration and the material costs of replacing component i. An ABM component can also fail before a SD. In this case, a CM action is necessary to replace the component. The costs and downtime of component i are c_i^{CM} and d_i^{CM}, respectively. The costs and downtime of CM are higher than for PM, which prefers PM over CM.

The age limit A_i of component i is the decision variable in the ABM model. This age limit should be interpreted in the following way during a SD:

- If the lifetime l of a component is smaller than the age limit A_i at a SD, then do nothing
- If the lifetime l is greater than or equal to the age limit $t \geq A_i$, then replace the component i preventively.

This behaviour of the model is represented in Figure 5.4. In the real-life situation at Lely, it could also happen that a component fails before the age limit A_i of a component. This can be included in the simulations model by simulating random failure moments from the Weibull distribution. However, when this occurs too frequently, a lower age limit is preferred to prevent these failures. If this is the case, the model also results in lower costs and downtime, as PM favours CM in both cost and downtime.

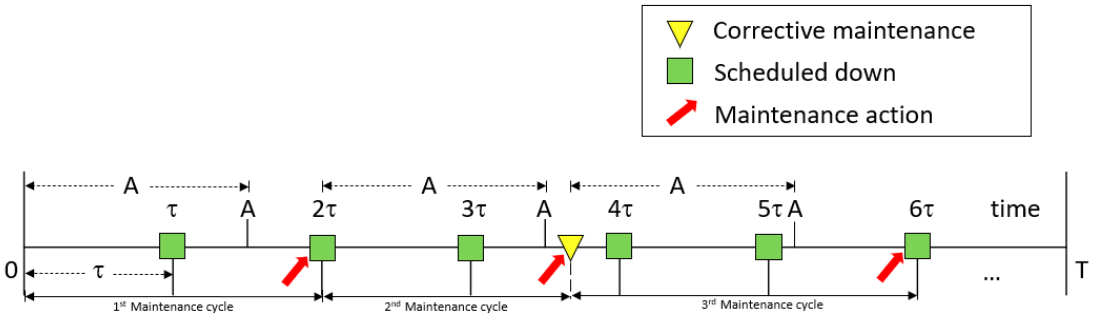

Figure 5.4: Age-based maintenance policy (adapted from Zhu (2015))

In the simulation model, the total costs for an ABM component i with age limit A_i in simulation trace n and maintenance interval τ is given by $K_{T_{i,n}}(\tau)$. These are the costs of all the CM and PM actions performed. The total downtime of component i with age limit A_i in trace n with a maintenance interval τ is given by $K_{T_{i,n}}(\tau)$. In an equation, this looks as follow:

$$K_{T_{i,n}}(\tau) = b * c_i^{CM} + o * c_i^{PM} \tag{5.13}$$

$$D_{T_{i,n}}(\tau) = b * d_i^{CM} + o * d_i^{PM} \tag{5.14}$$

The b stands for the number of component failures and o for the number of PM actions performed. The average yearly downtime and costs of component i with maintenance threshold A_i in simulation trace n and with maintenance interval τ are determined by dividing the total costs and downtime by the time horizon T in years, which is shown in the equations below.

$$Z_{T_{i,n}}(A_i(\tau)) = K_{T_{i,n}}(\tau) \;/\; T \tag{5.15}$$

$$V_{T_{i,n}}(A_i(\tau)) = D_{T_{i,n}}(\tau) \;/\; T \tag{5.16}$$

5.4 Failure-based components

Failure-based components are components that are only maintained when they fail. These components function until they fail, at which point they must be replaced. The maximum lifetime of a component is used with a FBM policy. A failure of a FBM component, on the other hand, always requires a CM action, which is more expensive and time-consuming than a PM action. Normally, electronic components and (sub)systems follow a FBM policy because electronics frequently have a constant or decreasing failure rate Arts (2017).

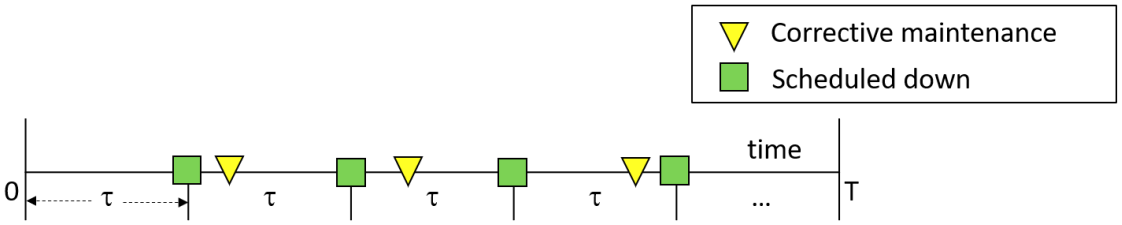

Figure 5.5: Failure based maintenance policy

In this research, FBM component i is assumed to fail according to a Weibull distribution with shape parameter β_i and scale parameter α_i. When a failure of component i occurs, corrective maintenance is required, which is visualized in Figure 5.5. The costs and downtime associated with a CM action of component i are c_i^{CM} and d_i^{CM} respectively. These costs and downtime consists of the costs and downtime of an USD, the labour costs, material costs and the duration of replacing component i. The total costs and downtime of component i in simulation trace n with maintenance interval τ are $K_{T_{i,n}}(\tau)$ and $D_{T_{i,n}}(\tau)$. These costs are determined by multiplying the number of CM performed (b) with the costs and downtime of CM for component i. In a formula, this looks as follow:

$$K_{T_{i,n}}(\tau) = b * c_i^{CM} \tag{5.17}$$

$$D_{T_{i,n}}(\tau) = b * d_i^{CM} \tag{5.18}$$

The average yearly costs and downtime of FBM component i in trace n are determined by dividing the total costs and downtime with time horizon T in years. This is shown in the equations below.

$$Z_{T_{i,n}}(\tau) = K_{T_{i,n}}(\tau) \ / \ T \tag{5.19}$$

$$V_{T_{i,n}}(\tau) = D_{T_{i,n}}(\tau) \ / \ T \tag{5.20}$$

Chapter 6

Multi-objective optimization

In this chapter, the multi-objective optimization problem in this study is highlighted. We have two different and contrasting objectives in our maintenance concept that must be optimized, the system downtime and the maintenance costs. To address this multi-objective optimization will be used. Section 6.1 describes how our problem fits in a Multi-objective optimization problem (MOOP). Different multi-objective optimization approaches according to the literature are explained in section 6.2. Section 6.3, 6.4 and section 6.5 explain three possible methods for optimization of multi-objective optimization problems.

6.1 The multi-objective optimization problem in our model

Our multi-component maintenance model is decomposed into individual component models. In these individual component models, the average yearly costs and downtime are determined by calculating the average of all simulation traces. The final average yearly costs and downtime of a CBM component i with control threshold C_i and a maintenance interval τ are given by $Z_{T_i}(C_i(\tau))$ and $V_{T_i}(C_i(\tau))$ respectively. The following multi-objective optimization problem exists for CBM component i for a particular maintenance interval τ:

$$\min_{C_i} \begin{bmatrix} Z_{T_i}(C_i) \\ V_{T_i}(C_i) \end{bmatrix} \tag{6.1}$$

subject to

$$0 \leq C_i \leq H_i \tag{6.2}$$

where $Z_{T_i}(C_i)$ and $V_{T_i}(C_i)$ are determined with Equations 5.3 and 5.6, respectively. The method used for the multi-objective optimization function of 6.1 is discussed later in this subsection. An ABM component's multi-objective optimization problem can be solved similarly. Only $Z_{T_i}(C_i)$, $V_{T_i}(C_i)$ and Equation 6.2 have to be substituted by $Z_{T_i}(A_i)$, $V_{T_i}(A_i)$ and $A_i \geq 0$, respectively.

Figure 6.1 shows the average yearly costs and downtime of CBM component i. The red line shows the average yearly costs and the blue line shows the average yearly downtime for the different control limit values C_i of component i. A simulation with 500 sets of random realized degradation steps, a time horizon of $T = 15$ and with a maintenance interval τ of 8 months creates this figure. The figure shows that the minimum value of the costs is obtained for a control threshold of 6, while the minimum value of the downtime is reached for a control threshold of 23.

When combining the information of both objectives and changing the axis of Figure 6.1, we obtain Figure 6.2. This Figure shows all component solutions as blue dots, including the Pareto solution in the grey box. A solution is Pareto optimal if no alternative solution improves one objective

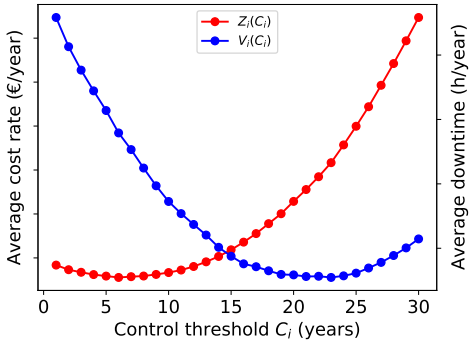

Figure 6.1: The average costs and downtime results of a CBM component with different control thresholds

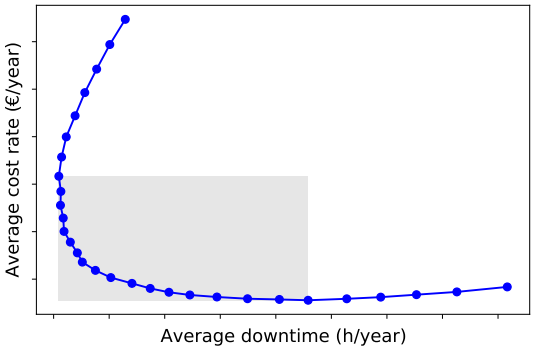

Figure 6.2: The Pareto frontier of a CBM component

without harming another objective (Marler and Arora, 2004). All Pareto optimal solutions in the grey box of Figure 6.2 together are the Pareto frontier. A trade-off between the two objectives is necessary to improve a solution. All solutions outside the grey box can always be improved in one objective without worsening the other objective. Every component solution from Figure 6.2 can be selected and used in a system solution. However, the Pareto frontier contains a set of the best solutions at a component level. Therefore, these solutions are considered when creating the system solution in this study. The CBM and ABM components in our study have a Pareto frontier which are considered by finding system solutions.

The system solutions are obtained by a summation of component solutions as shown in Equations 5.1 and 5.2. With single objective component solutions, there is one optimal solution per component. In our case, we deal with a multi-objective optimization on a component level, resulting in a Pareto frontier with Pareto optimal solutions of a component, as shown in Figure 6.2. All system solutions can be obtained by creating all possible combinations with the components solutions. As an example, we consider a system of three components where Component 1 and Component 2 have two Pareto solutions {1, 2} and Component 3 has three Pareto solutions {1, 2, 3}, the system solutions consists of all possible combinations of the component solutions: {(1, 1, 1), (1, 1, 2), (1, 1, 3), (1, 2, 1), (1, 2, 2), (1, 2, 3), (2, 1, 1), (2, 1, 2), (2, 1, 3), (2, 2, 1), (2, 2, 2), (2, 2, 3)}. In this example, we already have 12 different combinations at the system level. All combinations with the Pareto frontiers of component solutions have to be made to find the entire Pareto frontier at the system level. When considering a system with i components with p_i Pareto solutions and $i \in I_{ABM} \cup I_{CBM}$, the total number of Pareto component solutions is $\sum_i p_i$ which gives $\prod_i p_i$ system level combinations. The computational time will increase when the number of components and Pareto solution in the system increases.

6.2 Multi-objective optimization approaches

The two contrasting objectives, systems downtime and maintenance costs, are merged into a multi-objective framework in this study. The yearly average costs and average downtime are the two single objectives integrated into the framework for the multi-objective optimization to find the optimal trade-off balance between the two single objectives. According to Deb (2014), two approaches to solve a MOOP are the preference-based approach and the ideal approach. The preference-based approach corresponds to the prior articulation of the preferences category in the survey of Marler and Arora (2004). The ideal approach of Deb (2014) is in line with the "posteriori articulation of preferences" category of this survey.

The preference-based approach or prior articulation of preferences category assumes the use of higher-level information to combine the different objectives into a single composite function. This function is optimized using a single objective optimization method. In this method, weights are assigned to each objective to express the preference of the decision-maker. The ability to articulate preferences is critical to the success of the preference-based approach. In our case, we have two objectives that are hard to put weights on. It is difficult to translate the downtime into costs and vice versa in our situation. Therefore, the preference-based approach is not a suitable method to solve our multi-objective problem.

The second method of solving a MOOP is the ideal approach or posteriori articulation of preferences category. The idea behind this method is first to generate solution possibilities, after which one or several of the solution options are selected as the preferred solution(s) of the MOOP. This is also called the *generate-first-choose-later* approach (Messac and Mattson, 2002). In this method, the decision-maker can choose one or multiple solutions which he prefers.

A posteriori articulation of preferences approach is used in this research to solve the MOOP. Since it is challenging to define the a priori preferences for the objectives in our situation. With the posteriori approach, the decision-maker (Lely) can select the system solution(s) that they prefer. This approach also suits Lely's goal of individualizing the maintenance concept in their servitization project. Each farmer may choose their preferred trade-off option between downtime and costs, which suits best in their situation.

The remainder of this chapter explains in subsections 6.3, 6.4 and 6.5 three solution approaches for solving the MOOP of our multi-component maintenance model.

6.3 Weighted Sum Method

The most common approach used to solve a multi-objective optimization problem is the Weighted Sum Method (WSM) (Marler and Arora, 2004). All objectives are aggregated into one objective function by multiplying each objective with a weight ω. The idea of the WSM approach is simple. However, this approach introduces a new question. What are the values of the weights? This answer depends on a scaling factor and the importance of each objective. By normalizing the objective functions, the scaling effect can be reduced considerably. With the normalized objectives, the optimization problem can be transformed into a single-objective optimization problem by summing the normalized objectives with their weights. The normalization of both objectives can be calculated with the use of the following Equation (Marler and Arora, 2004):

$$F_m^{\text{norm}}(x) = \frac{F_m(\mathbf{x}) - F_m^{\circ}}{F_m^{\max} - F_m^{\circ}} \tag{6.3}$$

where $F_m^{\text{norm}}(x)$ is the normalized objective function. $F_m^{\max}$ is the maximum value of objective m and F_m° the ideal point of objective m, which is the best possible solution in a minimization problem of objective m. This can be translated into our situation with maintenance costs, downtime and control thresholds result in the normalization Equations 6.5 and 6.6 for costs and downtime for a given maintenance interval τ. It is common practice to choose the weight such that their sum equals one in the WSM (Deb, 2014). Therefore ω represents the weight with $0 \leq w \leq 1$. The single-objective optimization with the normalized objectives and weight ω looks as follow:

$$\min_{C_i} wZ_{T_i}^{\text{norm}}(C_i) + (1-w)V_{T_i}^{\text{norm}}(C_i) \tag{6.4}$$

subject to

$$Z_{T_i}^{\text{norm}}(C_i) = \frac{Z_{T_i}(C_i) - Z_{T_i}^{\min}}{Z_{T_i}^{\max} - Z_{T_i}^{\min}} \tag{6.5}$$

$$V_{T_i}^{\text{norm}}(C_i) = \frac{V_{T_i}(C_i) - V_{T_i}^{\min}}{V_{T_i}^{\max} - V_{T_i}^{\min}} \tag{6.6}$$

$$0 \leq C_i < H_i \tag{6.7}$$

The weight ω has a great influence on the final result in this approach. When we set the weight equal to one ($w = 1$), we will find the minimal cost system solution, which consists of all components solutions optimized on costs. The same logic applies to a weight of zero ($w = 0$), but in the case of minimal system downtime. The condition threshold (C_i) that minimized Equation 6.4 with a weight of ω is the component solution used in the system solution. This component solution is used to find the system costs and downtime in Equations 5.1 and 5.2 for a specific value of ω.

6.4 Weighted Sum with varying weights

With the WSM, all components are optimized with the same weight. However, there are situations when we prefer optimizing one component on costs and another component on downtime. To implement this, the weight ω from Equation 6.4 is replaced with ω_i. The ω_i represents the weight of component i in the optimization at component level in Equation 6.4. When the number of different weights of a component increases, the solutions space at the system level explodes. Hoedemakers (2020) used this method with two weights, $\omega_i = \{0, 1\}$ to solve his model. Combinations of the component solutions resulted in the system solutions where a component was either minimized on cost or downtime.

All different component combinations resulted in a large number of system solutions in the work of Hoedemakers. A Pareto filter was a recommendation by Hoedemakers (2020) to filter non-Pareto optimal solutions out of all solutions. Such a smart Pareto filter reduces all candidate solutions to only the best system solutions. The system solutions left after a Pareto filter will represent the Pareto frontier of the system solution based on minimal component cost or downtime solutions.

6.5 Genetic algorithm

Both approaches mentioned in previous sections solve the multi-objective optimization problem by using a single-objective optimization method. However, an approach as a Genetic Algorithm (GA) may be customized to solve a multi-objective problem directly. This subsection provides more information on a GA, after which there will be explained how a GA works. This is followed by several algorithms used for determining the fitness of a GA solution in multi-optimization problems.

John Holland was in 1975 the first one who used the term Genetic Algorithm, which is mainly abbreviated to GA. The term evolutionary algorithms (EAs) is also used by many people to cover the development in the last 15 years (Reeves, 2010). Genetic Algorithms are a type of optimization approach that has been used to solve problems in search, machine learning and optimization. A GA uses a search method based on smart randomization and a fitness function criterion to discover the best answers. Genetic Algorithm is frequently reported to be superior to other heuristics in the literature. Besides that, GAs effectively deal with very large problems and produce better outcomes than most other heuristic-based methods (Hussain and Sastry, 1997).

Phases in a Genetic Algorithm

A Genetic Algorithm (GA) is an approach that can be used to solve other multi-objective optimization problems. Figure 6.3 visualizes the different steps and the order of a GA. The phases are considered in a genetic algorithm are presented in Figure 6.3.

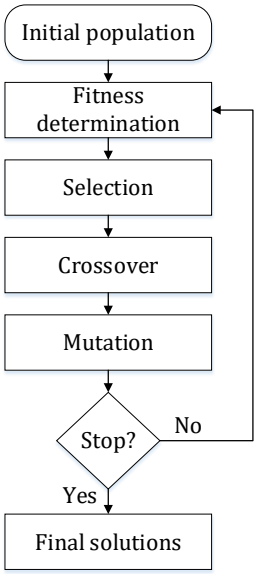

Figure 6.3: The considered steps in a Genetic Algorithm

These steps are discussed for a given value of τ in the remainder of this subsection. Only the ABM and CBM components are considered in the GA since only these components have Pareto optimal solutions for each τ.

Initial population
The process of a GA starts with creating an initial population. The population size is an input parameter of the GA. This initial population is the first generation and consists of multiple individual solutions for a given problem. Every individual in a population is called a chromosome and represents a system solution. This chromosome is characterized by a set of genes that represents the variables of the solution. In our case, the gene is the decision variable of the ABM or CBM components. Every gene indirectly represents an age limit (A_i) or control threshold (C_i) of a given component i. The Pareto optimal solutions of ABM and CBM component i are the possible values a gene can have.

As an example of how to link the Pareto optimal component solution to a gene, we look at the Pareto frontier of CBM component i from Figure 6.2. This Pareto frontier consists of eighteen Pareto solutions. The Pareto solution with the smallest control threshold (C_i=6) is numbered with 1 as gene value. The Pareto solution with the second-lowest control threshold gets number 2, and so on until the highest control threshold of the Pareto front is reached (in this example, C_i=23, which recieves a gene value of 18).

Figure 6.4 shows a population with four chromosomes. As can be seen, every chromosome consists of genes that can have different values. Every gene represents a numbered position in the Pareto frontier of a component. This numbered position is linked to a component maintenance threshold and the associating component costs and downtime for the maintenance threshold of this component. A chromosome with different genes represents one possible system solution. Only the numbered position in the Pareto frontier of ABM and CBM components are stored in a chromosome since only these components can have multiple Pareto optimal solutions. The system solution of a chromosome can be determined with Equations 5.1 and 5.2, where the numbers in the genes can find the ABM and CBM components solutions. Every system solution in our study consists of a certain value of system downtime and maintenance costs.

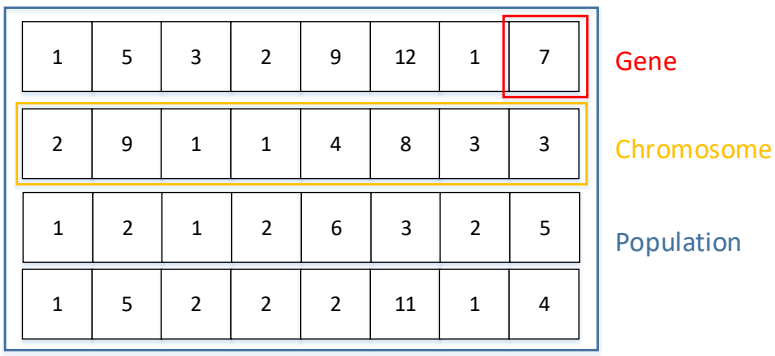

Figure 6.4: Population composition of a Genetic Algorithm

In short, every population consists of multiple chromosomes. Every chromosome represents a system solution with given age limits and control thresholds. The age limits and control thresholds of individual components are linked to the values of a gene in the chromosome. A chromosome presents a system solution with maintenance costs and system downtime for a particular combination of component solutions.

Fitness determination

The second phase in a Genetic Algorithm (GA) is the fitness function. In this step, the fitness of an individual solution is determined. In our situation, the fitness of an individual is determined based on the system downtime and maintenance costs. Every individual gets a fitness score representing how good the solution is in terms of downtime and costs. Individuals with better solutions (lower costs or downtime) have a higher chance of surviving their generation.

In this research, we have to deal with two objectives, system downtime and costs, whereby the fitness function must be formulated with respect to a trade-off between both objectives. The paper of Konak et al. (2006) is used as a starting point for finding a fitness function for a Multi-Objective Genetic Algorithm (MOGA). Three suitable fitness functions for a MOGA are discussed in the paragraphs below.

Random Weighed Genetic Algorithm (RWGA)

The first approach to access the fitness of solutions for a multi-objective optimization problem with a GA is the Random Weighed Genetic Algorithm (RWGA). In this MOGA approach, Murata et al. (1995) propose a weighted sum of the multiple objective functions. The fitness of each solution x with objectives m is determined with a normalized weight vector w_m, which is randomly generated for each solution. In this way, the fitness of every solution in each generation is determined with the randomly generated weight vector.

The approach of Murata et al. to assess the fitness looks similar to the weighted sum with varying weights of section 6.4. The major difference between the two approaches is the random selection of the weights instead of the predefined weights, as in section 6.4. The randomly generated weights in combinations with a GA ensure that not all different combinations are created. The selection phase of a GA ensures that the best individuals pass on to the next generation in the GA. Besides that, these best individuals are used as starting points for new individuals in a crossover. In this way, inferior solutions are filtered out of the GA solutions.

Strength Pareto Evolutionary Algorithm (SPEA)

The second considered approach for a MOGA utilizes a ranking technique to give non-dominated individuals a higher fitness ranking. This approach named as Strength Pareto Evolutionary Algo-

rithm (SPEA) and is proposed by Zitzler and Thiele (1999). A non-dominated individual is similar to a Pareto optimal solution. All solutions in a set of Pareto optimal solutions are non-dominated by the other solutions in this set (Konak et al., 2006).

The SPEA approach uses the ranking of non-dominated solutions to assign better fitness scores to the individuals. An external solution list called E stores the non-dominated solutions that have been found so far during the search. This list has a maximum capacity of X solutions. When the number of solutions in list E exceeds the maximum number of solutions X, a clustering algorithm reduces the number of solutions in list E.

The fitness of a population with solutions and the solution in list E is determined. First, the fitness of the non-dominated solutions from list E is determined. The following formula finds the strength of these non-dominated solutions: $s_i = \frac{n}{N+1}$, where n is the number of individual solutions that are worse than both objectives of non-dominated solution x and N is the number of all solutions. For the non-dominated solutions, the fitness of solution x is equal to the strength, $f_x = s_x$. The fitness of all dominated solutions is determined by one plus the sum of the strengths of all non-dominated solutions from list E, of which both objectives are superior.

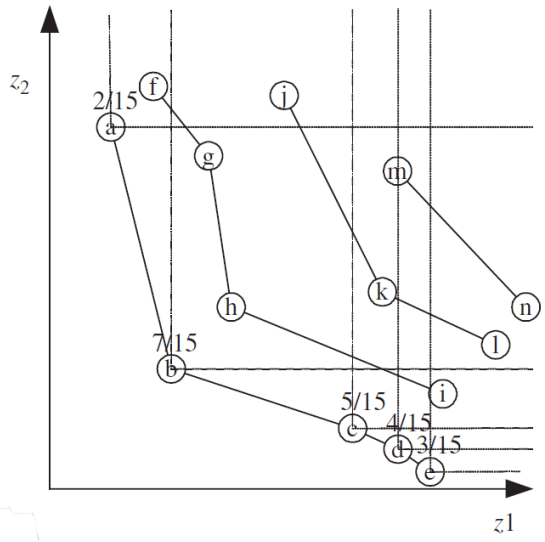

Figure 6.5: Fitness determination with SPEA method (from Konak et al. (2006))

This way of assessing fitness is represented in Figure 6.5. In this figure, the total number of solutions N is 14. When we look at solution A, we see that 2 solutions are worse than solution A on both objectives. Therefore the strength of solution A is equal to $s_A = \frac{2}{14+1} = \frac{2}{15}$. The same method is used for determining the strengths (and also the fitness) of the non-dominated solutions. As an example for determining the strength of a dominated solution, we look a solution J in Figure 6.5. The non-dominated solutions A and B are superior in both objectives for solution J. The fitness of solution J is therefore $1 + \frac{2}{15} + \frac{7}{15} = 1\frac{9}{15}$. The fitness of both list E and the dominated solutions are used as input for the selection procedure.

Fast Nondominated Sorting Genetic Algorithm (NSGA-II)
Another MOGA solving approach is the Fast Nondominated Sorting Genetic Algorithm (NSGA-II). This improved version of the Nondominated Sorting Genetic Algorithm (NSGA) approach is presented by Deb et al. (2002). The NSGA in a nutshell, first classifies a population into several non-dominated Pareto fronts. A dummy fitness value is given to all non-dominated Pareto fronts such that the solutions in front F_i with better solutions get a better fitness value than the solu-

tions in front F_{i+1}. In the NSGA method, all solutions in a given Pareto front receive the same fitness value. In Figure 6.6 solutions A to E get a better fitness than solution F until I because the solutions in front F_1 are better than those in front F_2 in this multi-objective minimization problem.

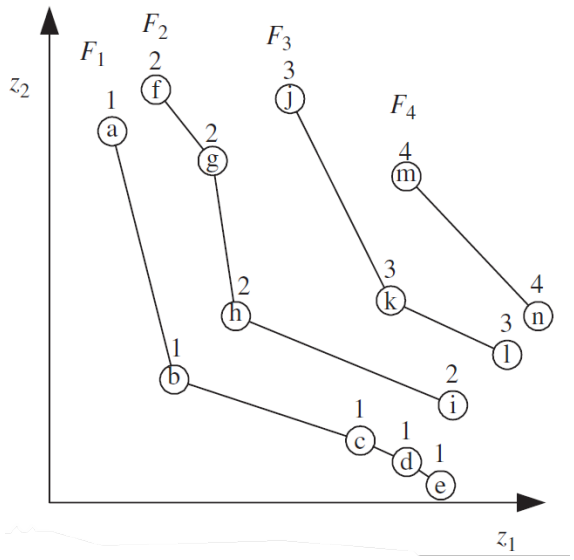

Figure 6.6: Fitness determination with NSGA method (from Konak et al. (2006))

As already mentioned, the NSGA-II is an improved version of the NSGA approach. The NSGA-II is improved by determining all individual solutions' fitness in a particular Pareto frontier. The fitness of the individual solutions in a Pareto frontier is determined with the crowding distance of a given solution i compared to all individuals of Pareto front I. The solution that is the closest to other solutions in Pareto front I receive the lowest scores.

The assessment of the fitness of the solutions in the Pareto frontier I starts with giving all solutions a crowding distance of 0. The second step is to sort all solutions on one objective. After that, the solution with the minimum or maximum value in an objective m should be found. In our study, we find the solutions in a Pareto frontier with the minimum cost and downtime. These two solutions receive a crowding distance of infinity. In Figure 6.6, solutions A and E get the infinity crowding distance in Pareto front F_1. For the other solutions in Pareto front I, the distance is determined by the sum of the crowding distance of all m objectives, $cd(x) = \sum_m cd_m(x)$. The crowding distance of solution x shows the distance between solution x and the nearest neighbours solutions. A low crowding distance indicates that the objectives of solutions are close to each other.

The crowding distance of objective m of solution x is determined with the use of the following formula:

$$cd_m\left(x_{[n,m]}\right) = \frac{z_m\left(x_{[n+1,m]}\right) - z_m\left(x_{[n-1,m]}\right)}{z_m^{\max} - z_m^{\min}}$$

where n is the position of solution x in the sorted solution list and z is the value of objective m. $z_m(x_{[n+1,m]})$ and $z_m(x_{[n-1,m]})$ represent the results of objective m from the solutions above and below the solution i in this sorted list. $z_m^{\max}$ and $z_m^{\min}$ represent the minimum and maximum values of objective m in Pareto frontier I.

The total fitness of the NSGA-II is determined by first sorting the solutions on their Pareto frontier number, followed by looking at the crowding distance. The individuals with a higher crowding

distance represent better solutions than individuals with a lower crowding distance when both solutions are in the same Pareto frontier. In Figure 6.6, solution B is getting a better fitness than solution D because the distance to other solutions of solution B is larger than for solution D. With this fitness, the individuals start the selection phase of a GA.

Selection
Based on the fitness score given to every single chromosome, a selection of the individuals is made. This selection consists of two sub-steps. First, the best n individual solutions of the population are selected. These n best individuals are sure to "*survive*" their generation. All other individuals are used as input in the *tournament selection*. The number of best individuals n is an input variable of the GA.

In the tournament selection, random individuals are selected from the population without the best n individuals. For example, in a tournament selection, two random individuals are chosen from the remaining population. These two selected individuals are compared based on their fitness scores. The selected individual with the best fitness is chosen and added to the population that passes to the next generation. The tournament size and number of randomly selected individuals are input parameters of the GA.

The result of the selection phase is a set with the n best individuals in a population. All winners of the tournament selection round are added to these best n individuals. The individuals in this set pass their genes to the next generation. This set is used as starting point of the next phase: the Crossover.

Crossover
In the crossover phase, new individuals named off-springs are created. The number of how many off-spring will be created is an input parameter of the GA. The crossover phase starts with randomly selection of two individuals from the set of individuals that passed the selection phase. These two selected individuals, which we call parents, are the starting point of the new individuals. The second step of the crossover is selecting a random place in the chromosome of the parents. This place is called the crossover point. The genes to the right of the crossover point from parent 1 are swapped with the genes to the right of the crossover point from parent 2. This results in two new individuals. These new individuals are called the off-springs. All newly created individuals from the crossover are saved in a set named offsprings.

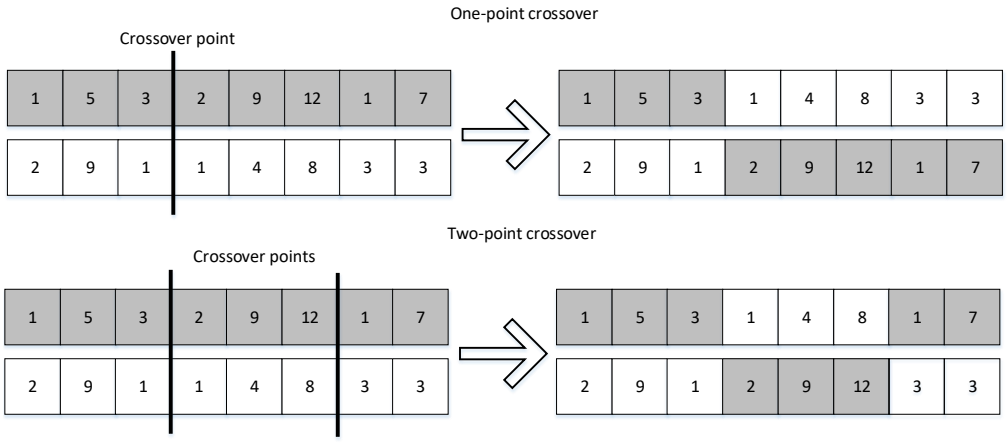

Figure 6.7: A one-point and a two-point crossover in a GA

Figure 6.7 shows a one-point and a two-point crossover. Next to the one-point and two-point crossover, there is also the k-point crossover where both parents' chromosomes have k crossover points. Another commonly used crossover that makes the crossover completely random is the uniform crossover. With this crossover method, every gene is randomly selected from one of the two parents with a uniform probability.

Mutation

The last phase of a GA is the mutation. In this phase, the new offspring individuals can be mutated with a low random probability. This implies that the genes of a new offspring can be changed to another value. How low the probability of a mutation is, is a decision variable of the GA and the type of mutation. Figure 6.8 shows examples used mutation types in a GA.

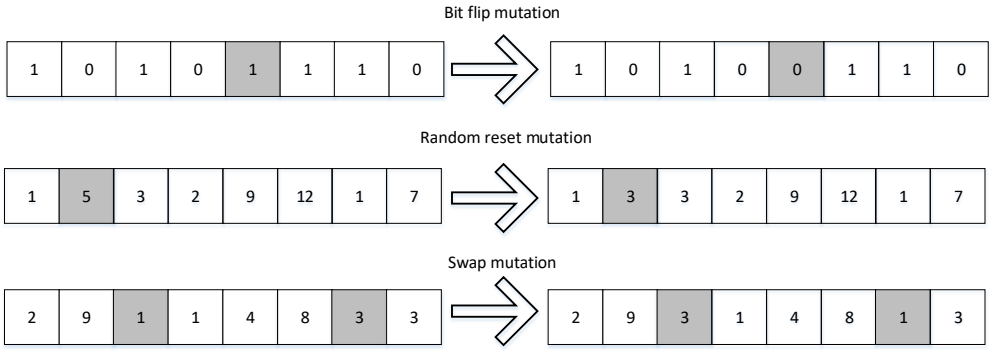

Figure 6.8: Mutation types used in a Genetic Algorithm

In the flip bit mutation, the value of a gene is randomly flipped. This mutation method is used for binary-encoded GAs. The adapted version of the flip bit mutation suitable for integer values is the random resetting. The same principle as at the flip bit mutation occurs, but a random value from a predefined range is selected instead of the flip of a bit. The last mutation method that we discuss is the swap mutation. In this mutation, two randomly selected genes are swapped in a chromosome.

The above-described steps are repeated until a Genetic Algorithm reaches its maximum number of generations. The objective function has reached a pre-defined value, or no improvements are made in the population in a predefined number of iterations.

Chapter 7

Parameters estimations

In this chapter, the methods for estimating the parameters used as input for the maintenance model are explained. The estimation of the Weibull distribution parameters that are used to model the lifetime of ABM and FBM components is provided in this chapter. Additionally, the parameters for the Gamma process are estimated in this chapter which is used to model the degradation of the CBM components. In section 7.1, the Bayesian updating method for updating the parameters of the Weibull distribution with censored service data of ABM and FBM components are described. Section 7.2 represents the method used to estimate the Gamma process parameters for CBM components by fitting a Gamma distribution on component degradation based on service data. The newly found failure and degradation parameters replace the older parameters in the maintenance model.

7.1 Updating of failure predictions

A Bayesian updating mechanism is a suitable approach to updating uncertain parameters in a distribution (Beck and Katafygiotis, 1998). The parameters of a failure distribution are reassessed and updated with newly available information in this mechanism. By updating a maintenance model with new data, the new model solution likely gives a better impression of the real situation. This section third research question is answered:

3. *How to include updating of components failure information based on new evidence in a maintenance concept?*

This section is divided into two parts. The literature consulted on Bayesian updating of a Weibull distribution using censored lifetime data is provided in the first part of this section. The Bayesian updating method that is applied in this research is covered in the second part of this section.

7.1.1 Bayesian update methods

In this section, the Bayes' theorem is briefly introduced as this forms the basis for the Bayesian updating mechanisms used in the literature on maintenance modelling and optimization. This is followed by several papers from the literature reviewed to obtain more knowledge on Bayesian updating and its application in a maintenance model.

Bayesian updating of distribution parameters is based on the Bayes theorem. The Bayes theorem is a probability rule that expresses the likelihood that a particular event will occur expressed in the conditional probabilities of all possible events. The Bayes theorem is as follow:

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)} = \frac{P(B \mid A)P(A)}{P(B)} \tag{7.1}$$

Event B can happen with or without the occurrence of event A. From the conditional probabilities of B given probabilities that A happens or not, the probability that A happens when knowing that B happened is shown in Equation 7.1.

This theorem can also be applied to distributions and data. First, we guess a distribution that is called a prior distribution. After guessing the prior distribution, we discover some data representing the true distribution. Given the discovered data, we can determine the likelihood that the discovered data fits the prior distribution. The likelihood that the data suits the prior distribution multiplied by the prior distribution gives the posterior distribution.

A Bayesian updating method for a new CBM policy in complex systems based on the status of the system components is investigated by Walter and Flapper (2017). The component status in this policy is either working or defective. All components failure distributions in this policy are modelled by a Weibull distribution with an iteratively updated scale and a fixed shape parameter. A weighted average updating rule is used for updating the scale parameter. Right-censored observations and non-censored lifetime observations can be used for updating the scale parameters for all components in the system. In our study, we also have to deal with censored and non-censored lifetime observations.

Shi et al. (2020) developed a condition-based system reliability decision-framework for multi-component systems with Bayesian updating. A Bayesian updating method is used to improve the estimation of the component degradation parameters. Shi et al. (2020) assumed hidden components' failures that can only be detected during physical inspections of the system. In this work, Shi et al. (2020) use predictions of component readabilities to compute the reliability of the sub-systems and the system.

Drent et al. (2020) propose a Bayesian policy for age-based replaced components with initially unknown lifetime distribution parameters. The commonly used newsboy distributions in inventory research were adopted and used to investigate age replacement decisions from censored and uncensored data. A newly established stochastic property made this framework useful for age-related maintenance problems. An example found the optimal age replacement interval with Weibull distributed component lifetime after roughly 500 corrective or preventive component replacements.

A Bayesian updating approach for both parameters of the Weibull distribution is proposed by Soland (1969). The scale, as well as the shape parameter, are treated as unknown in this study. A Gamma distribution models the scale parameters and the shape parameter is modeled with deterministic probabilities. Besides that, censored data and non-censored data are used to update both parameters. A Monte Carlo simulation in combination with numerical integration is used to perform the updating procedure. An example with censored data showed the posterior probabilities of the shape parameter and the posterior parameters that model the Weibull scale parameter.

Kundu (2008) applied Bayesian updating of unknown parameters with censored data. The updating is performed with a known and an unknown shape parameter. With a known shape parameter, the unknown scale parameter is modelled by a Gamma distribution. When both parameters are unknown, the Weibull scale and shape parameter is assumed to have a joint prior distribution. Monte Carlo simulations are performed to compute the updated values of the unknown parameters.

7.1.2 Applied updating method

In this research, a Bayesian updating approach of the Weibull scale parameter is used. We assume to have a fixed value for the shape parameter. The parameters of the Weibull distribution that Hoedemakers (2020) used in this research are useful in our Bayesian updating method as a prior parameter. Hoedemakers conducted his research about a multi-component maintenance model for systems new to the market on the same Collector at Lely. Therefore, these parameters are suitable

to use in the Bayesian updating part of this study. The scale and shape parameters of component i used from Hoedemakers (2020) are denoted by $\alpha_{i,HD}$ and $\beta_{i,HD}$, respectively. Estimating the failure behaviour of new products often contains uncertainty. Therefore, we assume that both the prior scale and prior shape parameters follow a Uniform distribution. The Uniform distribution of both prior distributions of component i are presented below:

$$\text{prior } \alpha_i \sim Uniform(\alpha_{i,HD} - 1, \alpha_{i,HD} + 1) \tag{7.2}$$

$$\text{prior } \beta_i \sim Uniform(\beta_{i,HD} - 1, \beta_{i,HD} + 1) \tag{7.3}$$

Both the censored and failure data will be used as input data in the Bayesian updating method. The implementation of the Bayesian updating is performed in Python Monte Carlo simulations and the use of the PyMC3[1] package. An updating model with the above-mentioned prior parameters is created with the use of this package. The Bayesian updating model is verified in Appendix A using data sets with censored data from Ebeling (2019).

The Monte Carlo simulation used for the Bayesian updating has a number of simulation runs as an input variable. The number of simulation runs shows the number of independent simulations used for the parameter updating. A higher number of simulations increases the accuracy of the Monte Carlo simulation and increases the computational time. For computational issues, we use 1000 runs as input variables in our Monte Carlo simulation. The average updated parameter of these 1000 updating results is used as the posterior parameter in our study.

7.2 Gamma process determination

In this section, the method applied for determining the degradation parameters of the Gamma process is described. To find the parameters of a Gamma process, the degradation information of a component is required. Unfortunately, this information is not available. This means that we could only estimate the component degradation by making assumptions and the available service data. The processed data from section 3.3 is used as a starting point to determine the parameters for the Gamma process. In the service data, PM actions of CBM components are sometimes filed as repairs (CM). Before starting the estimation of the Gamma process parameters, this must be adjusted first.

A preventive replacement action of CBM components filled as a repair in the service data can only happen after a short period from a PM visit. During a PM, the service engineer found that the condition of the CBM is below the condition threshold, which means that a preventive replacement is required. To find all CBM component actions that are filled as PM instead of CM, there is assumed that the repair action happens within a week after the PM visit. The number of PM actions in the service data increased from 1013 to 1106 PM actions by correcting this in the data. The parameter estimation of the Gamma process can start with the service data, including this corrected action.

With the available service data, it is possible to determine the runtime of a component (see section 3.3.2). The duration information in combination with an assumed degradation for CM and PM action gives us the degradation rate of a replaced component. When this is applied to the service data for component i, we get a list of degradation rates of replaced components i. The assumed component degradation at a CM action is larger than during a PM action.

The list with degradation rates of the replaced components is rescaled to find the degradation of a replaced component per year. This list with the degradation of a component per year is used to fit a Gamma distribution with the use of the Maximum Likelihood Estimation (MLE). The

[1] https://docs.pymc.io/

MLE is a widely used approach for estimating the parameters of a distribution. At the MLE, the maximum likelihood function is maximized by taking the derivative of the estimated parameters and equal this to zero. The MLE is a safe method to estimate distribution parameters even with small sample sizes, according to Genschel and Meeker (2010). In this study, we have to deal with a small to a medium amount of data. Therefore, the MLE technique is used in this research.

We used a distribution fit on all available processed service data instead of the Bayesian updating approach for finding the new degradation parameters. The use of service data is an unusual starting point for finding degradation parameters. Therefore, we decided to apply a distribution fit on all available process service dates instead of Bayesian updating with only newly collected data. The parameters of the Gamma process are replaced by new parameters found by a distribution fit on all available service data after the transformation to degradation data. With this approach, we fit a distribution to complete data, including the newly available data.

Chapter 8

Case study

This chapter applies the developed modelling approach for a multi-component maintenance model in a finite horizon to the Collector. The results of this case study are presented in this chapter. Additionally, a scenario analysis is performed to see the effects of different input values in the model. These scenario analyses are performed for uncertain input parameters of our case study.

8.1 Model assumption

The model used in this research is based on several assumptions. The assumptions made for this model are listed in section 5.1. The multi-component maintenance model created in Chapter 5 is based on these assumptions.

This model is based on the model of Zhu (2015), who only incorporates positive economic dependencies between systems' components. In the case study for the Collector also structural and stochastic dependencies should be taken into account. Component E and another component have a stochastic dependency in the Collector. If Component E breaks, the lifespan of another component may be reduced. We assume that if Component E fails, another additional component is also damaged to account for this dependency. The replacement of the expensive additional component during a CM action of Component E results in higher CM costs. Component F and G have the same dependencies as the additional component and Component E. As a result of the higher CM costs, a more conservative maintenance threshold will be achieved for these components.

The case study in this chapter focuses on the Collector, as in Hoedemakers (2020). Since the way of modelling the system with the individual components is comparable in both studies, the same assumptions for individual components are applied in this study. These assumptions are:

- Every Collector consists of two Components B. Both Components B are replaced when one has failed or is worn out. Therefore, Component B is modelled as one composite component.
- Component H is part of Component N. If Component N is replaced, also a new Component H is placed. It is possible to replace Component H without replacing Component N. Therefore, we assume two independent components in the model, which leads to higher cost than in reality.
- Failures of Component E are normally hidden failures that our model is not able to deal with. Therefore, we assume that a failure of Component E results in an USD resulting in higher costs in the model than in practice.
- Visual inspections are currently performed for maintaining Component M. The replacement depends on the interpretation of a service engineer without a strict failure threshold. This makes it hard to model Component M as a CBM component. Therefore, Component M is modelled as ABM components.

Additionally to above assumptions, we assume that all ABM and FBM components in the system fail according to a Weibull distribution. Fitting a Weibull distribution on lifetime data is a common approach at Lely. These Weibull fits suited well on the lifetime data. Besides that, the Weibull distribution is much used for modelling component lifetimes because of the good fit with data (Arts, 2017). For the degradation of CBM in the Collector, we assume that this can be modelled with a Gamma process. For both the distribution to model the failure behaviour and the degradation, we assume stationary distributions.

8.2 Input parameters

In this section, the parameters that are used as input in the simulation model are presented. In section 8.2.1, the approach to derive the cost and downtime input parameters is explained. The actual costs and downtime input parameters of individual components are presented in sections 8.2.2, 8.2.3 and 8.2.4. In these sections, the input parameters of the Collector components are grouped by the maintenance types. The failure and degradation parameters of the Collector components are determined using the cleaned and prepared data described in Section 3.3.

8.2.1 Cost and downtime parameters

The costs and downtime corresponding to the maintenance performed of a Collector are derived from a farmer's perspective. This is because Lely intends to offer personalized maintenance concepts for individual farmers in the future. With an individualized maintenance concept, every farmer can choose a maintenance concept that suits their needs based on system downtime and system costs. Therefore the commercial prices for the labor of a technician and the recommended retail prices of spare parts are used.

This study is a continuation of the study from Hoedemakers (2020) that also focused on the Collector at Lely. Since Hoedemakers applied a good method of derivation the costs and downtime for a component, the same method is used in this study. Next to the same cost and downtime derivation method, the same input values from Hoedemakers (2020) are used in this research. Our study was performed a year later; therefore, it is reasonable to assume that the costs and downtime of components are currently similar to that of last year. For completeness, the derivation of the input costs and downtime is presented below:

- d^{SD} is the setup downtime of a SD. This is the downtime during a PM action that a service engineer needs to perform his routine tasks. These tasks take 15 minutes which is equal to 0.25 hours as d^{SD}.
- c^{SD} is the setup costs during a SD. These costs include two times the driving time $t^{drivingSD}$ to a farm and the time needed to perform the SD tasks. We assume that the driving time $t^{drivingSD}$ for a PM is approximately 25 minutes. The commercial price of labour for a service engineer is given by $c^{engineer}$. All together is results in the following setup costs of a SD:

$$c^{SD} = c^{engineer}\left(2 \times t^{drivingSD} + d^{S-SD}\right) = 62(2 \times 0.42 + 0.25) = €66.94$$

- d^{USD} represents the setup downtime of an USD. This setup downtime consists of the average time to react, the one-way drive time, and an intake with the farmer. During this time, the system does not work. The average react time t^{react} varies per Lely Center and depends on whether a service engineer has to finish another job first. The total setup downtime of an USD is the sum of the estimates for the t^{react}, the driving time for an USD $t^{drivingUSD}$ and time of an intake, t^{intake}

$$d^{USD} = t^{react} + t^{drivingUSD} + t^{intake}$$

- c^{USD} is the setup cost for an USD. This is determined by two times the $t^{drivingUSD}$ plus the time for an intake t^{intake}.

$$c^{S-USD} = c^{engineer} \left(2 \times t^{drivingUSD} + t^{intake}\right)$$

- d_i^{PM} is the downtime for a preventive replacement of component i. In our study, this is equal to the replacement time of component i, d_i^{PM}=$t_i^{replacement}$.
- d_i^{CM} is the downtime of a CM action of component i. If the failure of component i does not cause other failures, d_i^{CM} is equal to:

$$d_i^{CM} = t_i^{replacement} + d^{USD}$$

  Failures of components can also require the replacement of other components. In this case, additional downtime of damage by component i is given by $d_i^{additional}$. The total downtime in this the situation is given by:

$$d_i^{CM} = t_i^{replacement} + d_i^{additional} + d^{USD}$$

- c_i^{PM} is the costs of preventive maintenance of component i. These costs are equal to the replacement duration multiplied with the labour costs of a service engineer plus the costs of part i. This results in:

$$c_i^{PM} = t_i^{replacement} \times c^{engineer} + c_i^{part}$$

- c_i^{CM} represents the cost CM for component i. These costs are given by:

$$c_i^{CM} = c_i^{PM} + c^{USD}$$

  Here component i does not cause additional damages to other components. The CM costs of components with additional damages are given by:

$$c_i^{CM} = c_i^{PM} + c_i^{additional} + c^{USD}$$

  where $c_i^{additional}$ represents the additional labour and part costs for the additional damage.

8.2.2 CBM components

The Collector that we model in this case study has four CBM components: Component A, Component B, Component C and Component D. The degradation level of a CBM component is continuously measured in the model of Zhu (2015). In our study, we only obtain the degradation level of CBM components during a visual inspection at a SD. Obtaining information of component degradation during a SD should not give problems for using the CBM model of Zhu (2015) because PM actions are only determined based on the control limit during a SD. With the following assumptions, we are still able to use the CBM model of Zhu:

i The inspection of CBM components occurs during very SD. As a result, the costs and downtime of these inspections can be incorporated into the setup costs and downtime of a SD.

ii There are always enough spare parts available during a SD to replace a CBM component when the condition is below the control limit, $X_i(t) \leq C_i$.

The input parameters of condition-based components are shown in Table 8.1. The second column shows the initial degradation level of new components. As can be seen from Table 8.1, all new placed components do not have any degradation. The third column shows the failure thresholds

of the CBM components. If a component reaches this degradation level, it fails and a CM action is required. In the case of Component A, the initial degradation of 0 intends an initial condition of 100. Besides that, we assume that Component A fails when it reaches a condition of 65, which is at a degradation level of 35. Component B has an initial condition of 30 and an initial degradation level of 0. This component fails when the condition level is 0, which is equal to a degradation of 30. Component C and D have an initial condition of 65 and they fail after wear of 15, which is at a condition level of 50. The model of Zhu assumes monotonic increasing degradation levels, which is the reason for these transformations. The current profile dept is much more intuitive to measure than the degradation level. For this reason, the later presented results show the condition level of a component instead of the degradation level.

Table 8.1: Input parameters simulation model of condition-based components

Component name	**Initial level** (x_i)	**Failure threshold** (H_i)	d_i^{PM} (h)	d_i^{CM} (h)	c_i^{PM} (€)	c_i^{CM} (€)
Component A	0	35	0.08	0.94	67.34	143.95
Component B	0	30	0.16	1.02	140.48	217.10
Component C	0	15	0.11	0.97	21.89	98.50
Component D	0	15	0.11	0.97	48.89	125.50

The degradation of the CBM components is modelled with a Gamma process. Parameter $\gamma_{i,0}$ in Table 8.2 represents the shape parameter of component i at the start of the model. The initial Gamma process parameters are found by simulating 10.000 random data points with the degradation input parameters of the RCM from Hoedemakers (2020). A Gamma distribution is fitted with the MLE on these randomly created data points to find the corresponding parameters of the Gamma process.

Table 8.2: Degradation parameters simulation model of condition-based components

Component name	**Gamma shape**					**Gamma scale**				
	$\gamma_{i,0}$	$\gamma_{i,1}$	$\gamma_{i,2}$	$\gamma_{i,3}$	$\gamma_{i,4}$	$\eta_{i,0}$	$\eta_{i,1}$	$\eta_{i,2}$	$\eta_{i,3}$	$\eta_{i,4}$
Component A	8.94	1.23	1.85	2.37	2.10	0.97	51.46	29.63	19.12	17.33
Component B	4.04	4.93	2.61	2.02	1.95	6.87	2.42	8.24	14.31	15.55
Component C	7.75	7.31	2.21	3.47	3.72	3.16	2.14	8.41	5.63	5.23
Component D	7.96	3.60	2.63	3.09	2.97	2.13	3.43	5.69	5.92	5.48

All degradation parameters of the CBM components are presented in Table 8.2. As can be seen in this table, there are five different shape and scale parameters for every component. These degradation parameters are replaced by new degradation parameters based on new service available data. The method of finding the new parameters for the Gamma process is described in section 7.2. The new input parameters of the Gamma process are used to model the degradation of a component in different moments in the simulation model.

In our simulation model, we use four updating moments. The initial parameters $\gamma_{i,0}$ and $\eta_{i,0}$ are used to model the degradation for the first period until new degradation parameters are found. The new degradation parameters of the Gamma process are determined with intervals of one year in our study. The random degradation of component i is modelled with parameters $\gamma_{i,0}$ and $\eta_{i,0}$ in the first year and with $\gamma_{i,1}$ and $\eta_{i,1}$ in the second year. This continues until parameters $\gamma_{i,4}$ and $\eta_{i,4}$ are found at the end of year four. From this moment till the end of simulation horizon T, the random component degradation is modelled by a Gamma process with $\gamma_{i,4}$ and $\eta_{i,4}$ as parameters.

The shape parameters from table 8.2 model the degradation of a component per year. This parameter can be transformed to find the degradation of a component for a given time period by

multiplying γ with Δt. The Δt represents the time in years to find a component degradation. To find a random degradation of a component per three months, Δt is equal to 0.25. In our study, we use monthly random degradation increases. Therefore, we use a Δt of 1/12 years.

8.2.3 ABM components

The input parameters used in the simulation model for age-based components are listed in Table 8.3. The second column of this table shows the Weibull shape parameters of the ABM components. These values are found with a Uniform distribution of Equation 7.3 based on the shape parameters found by Hoedemakers (2020). Column three to column seven show the different values of the Weibull scale parameter. The values of $\alpha_{i,0}$ show the initial scale parameters found by a Uniform distribution of Equation 7.2 based on Hoedemakers scale parameters. The Weibull scale parameters in column four to seven show the updated scale parameters found by Bayesian updating with updating periods of one year.

Table 8.3: Input parameters simulation model of age-based components

Component name	**Weibull shape**	**Weibull scales**					d_i^{PM}	d_i^{CM}	c_i^{PM}	c_i^{CM}
	β_i	$\alpha_{i,0}$	$\alpha_{i,1}$	$\alpha_{i,2}$	$\alpha_{i,3}$	$\alpha_{i,4}$	(h)	(h)	(€)	(€)
Component E (2)	3.20	1.56	4.94	7.77	10.56	9.13	0.14	1.18	20.27	739.27
Component F (2)	3.05	10.52	10.71	11.29	12.92	15.75	0.19	1.18	36.64	739.27
Component G (2)	3.04	10.16	10.36	11.02	12.67	15.58	0.35	1.18	29.74	739.27
Component H	3.20	3.90	6.22	8.49	11.00	8.43	0.16	1.02	28.60	105.21
Component I	2.87	2.32	5.35	6.72	5.93	6.72	0.03	0.89	22.69	99.30
Component J	1.86	4.67	4.37	6.16	7.06	9.54	0.03	0.89	6.28	82.89
Component K	1.75	4.91	4.6	6.56	7.79	10.37	0.03	0.89	6.28	82.89
Component L (2)	1.56	6.71	8.33	10.33	13.04	14.31	0.16	1.02	72.75	149.53
Component M	2.55	4.99	5.22	3.39	4.97	9.35	0.08	0.94	51.07	127.69

We use random realizations of failure moments simulated from a Weibull distribution in our simulation. In a simulation with all four updating moments, the failures in the first year are generated by a Weibull distribution with shape β_i and scale $\alpha_{i,0}$. In the second year, the failure moments are based on the shape β_i and scale $\alpha_{i,1}$, until scale $\alpha_{i,4}$ is reached by the end of year four. From this moment, the failure moments in the remaining simulation years of the horizon T are simulated with a Weibull distribution with shape β_i and scale $\alpha_{i,4}$.

There are multiple components in the Collector from which two parts are used in the system. The components which occur more than once in the system are labelled with (2) in Table 8.3. In the model, these components are independent of each other, allowing a separate replacement of these components.

8.2.4 FBM components

Table 8.4 shows the input parameters of FBM components in our simulation model. Two FBM components are critical for the functioning of the Collector. When one of these components fails, the complete system is down. An USD including a replacement of the failed component is triggered by a failure of one of the FBM components.

The modelling of failure moments realization with the updated scale parameter α_i is similar to ABM components. In a simulation with all updating moments, the realized failure moments in the first year are model with $\alpha_{i,0}$, and with $\alpha_{i,1}$ in the second year. This continues till $\alpha_{i,4}$ is reached at the end of the fourth year. From this moment, all the failure moments in the remaining time horizon are simulated by a Weibull distribution with a scale of $\alpha_{i,4}$ and shape of β_i.

Table 8.4: Input parameters simulation model of failure-based components

Component name	**Weibull shape**	**Weibull scales**					d_i^{CM}	c_i^{CM}
	β_i	$\alpha_{i,0}$	$\alpha_{i,1}$	$\alpha_{i,2}$	$\alpha_{i,3}$	$\alpha_{i,4}$	(h)	(€)
Component N	1.07	10.15	8.52	3.89	6.33	11.11	0.94	209.32
Component O	1.39	11.19	11.75	13.51	15.34	19.02	0.89	97.24

8.3 Simulation model

In this section, the simulation approach that is used to solve this model is explained. In the first part of this section, the simulation models of CBM, ABM and FBM components are presented. Furthermore, the comparing method individual component results with different decision variables is given in this section. In the second part of this section, the Bayesian updating procedure for updating the failure parameters in the simulation is clarified.

This research aims to create a multi-component maintenance model with a time horizon that is suitable for Lely's situation. From Chapter 4 we concluded that a model with a finite horizon is preferred because of the non-stationary composition of system components and the use of newly available service data. Chapter 5 provides the mathematical formulations of a decomposed system with multi-objectives with a finite horizon T. This time horizon is selected as the expected lifetime of 15 years for the Collector, which is determined by experts from Lely. The maintenance interval τ is the decision variable at a system level and the maintenance thresholds are the decision variables of CBM and ABM components in the system. The application of a MOOP to our multi-component model is explained in section 6.1. To solve a multi-objective optimization at the system level, we first have to find solutions at the component level.

8.3.1 Component simulation

The simulation method applied for finding solutions of CBM components is described in this section. As explained in section 5.2, the degradation of CBM increases over time. In our model, we assume that a Gamma process models the increase of degradation (increment) of a CBM. Our simulation model models the increments of component i with random realizations of a Gamma process with shape parameter $\gamma_i \Delta t$ and scale parameter η_i. For the Δt in this model holds $\Delta t \leq \tau$. The multiple degradation increments occur between two maintenance visits in the simulation model.

In the simulation model, random increments are added to the component condition to keep track of the degradation level of component i. If the degradation level $X_i(t)$ is above the maintenance threshold C_i during a SD, the component is replaced preventive. At this moment, the degradation level $X_i(t)$ is set to the initial degradation level of a new component. Besides that, the costs and downtime of a PM action for component i are saved in the simulation.

When $X_i(t)$ is below the maintenance threshold C_i during a SD in the simulation, nothing happens. The simulation continues by adding new increments to degradation level $X_i(t)$. If at some point the degradation level of the component reaches the failure threshold H_i, the component fails and a CM action is performed. The degradation level is set to the initial degradation x_i of component i after a corrective replacement. The costs and downtime of a CM action are also saved in the simulation. All the costs and downtime in a simulation are saved where-after the average yearly costs and downtime determined with the use of Equation 5.9 till Equation 5.12.

To compare simulation results of component i with different maintenance thresholds C_i and maintenance intervals τ multiple simulations are required. The variability and the number of simulations can be reduced when simulations with different decision variables use the same realized

degradation input. A set of realized degradation is used in simulations with all different decision variables. In the example of a CBM component, random increments are realized and stored in a set. This set with realized increments is used in all simulations with different decision variables for component i and the system once. In an example where component i has 10 options as maintenance thresholds and the system considers 5 different values of τ, the same set with realized increments is used in these 50 simulation scenarios once.

The same idea of realized increments can be applied to ABM and FBM components models. For ABM and FBM components, failure moments from a Weibull distribution are simulated instead of increments for CBM components. The combination of a set with realizations for all components $i \in I$ is named a trace in the simulation model. The number of traces in a simulation represents the number of simulations with different realized degradation and failure sets for all system components.

As a result of using traces instead of random data in a simulation, fewer simulation traces are required to compare the results with different decision variables. The results with different decision variables can be easier compared when using the same realized increments and failure moments as input. This results in less variability in the output of the different simulation scenarios since the same input is used.

8.3.2 Updating implementation

In our maintenance model, we aim to create a model that is close to reality. This research aims to model the maintenance of a new system in the first years after its introduction on the market. In the begin mostly expert knowledge and test data of a new system are available. This information can be used to model the system the first period after the introduction on the market. After introducing of a new system to the market, new data becomes available on how the system behaves and fails. The initial information from experts or test data can be updated using newly available data and Bayesian updating. These updates based on new information can be incorporated into the simulation model to create a model that comes close to reality. The Bayesian updating can be applied at different moments in time $t_1, t_2, ..., t_u$ where $t_u < T$, with T as the model time horizon in years.

The modelling of a system starts after the introduction to the market at t_0. At this moment, the system is modelled based on the expert knowledge or test data for the complete time horizon T. In our situation, we use a Uniform distribution of the component lifetime parameters that Hoedemakers (2020) used in his study to model a just launched Collector. After the launch of a new machine, new information about the systems failure behaviour becomes available. At a moment in time t_1, the initial component lifetime parameters are updated with newly available information. After the updating at time t_1, the system will be modelled with the updated parameters to find the system solution.

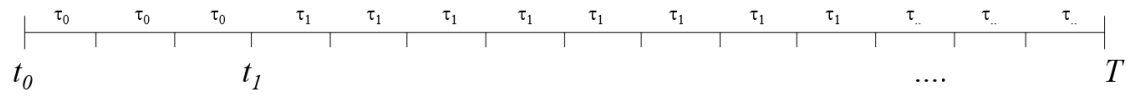

Figure 8.1: Bayesian updating implementation in the simulation model

Figure 8.1 shows the implementation of the Bayesian updated failure parameters in the simulation model. The system is modelled with the initial parameters of the introduction to the market at t_0 until updating moment t_1. In this period, the system is maintained according to the maintenance decisions selected from the maintenance model at t_0. From time t_1 till the end of time horizon T, the system should be modelled with the newly updated parameters. The same modelling approach is applicable for a new system starting at t_0 with multiple parameter updating moments.

Figure 8.2 presents the timeline of a simulation model with three updating moments. Updating parameters could result in other maintenance thresholds at the component level or even to other maintenance intervals at the system level. For example, at t_0 a different maintenance threshold could result in lower downtime for component i than at t_2. Another example is the maintenance interval of 0.5 years (6 months) gives minimal maintenance costs at time t_1, while a maintenance interval of 0.75 years (9 months) is cost-optimal at time t_3.

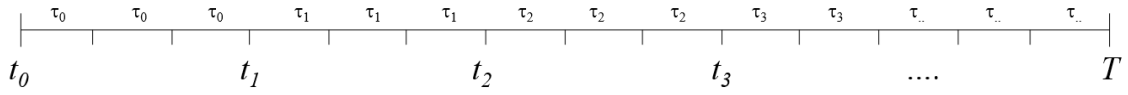

Figure 8.2: Bayesian updating with several updating moments in the simulation model

The updated failure parameters for ABM and FBM and new degradation parameters for CBM components are implemented in our simulation model. The degradation of CBM components after update moment t_1 is modelled by the Gamma process parameters found at time t_1. The same applies to ABM and FBM components. New failure moments later than t_1 are model with the Weibull parameters found at t_1. The first simulation failure moment with parameters found at t_1 is used after a component replacement later than t_1. The updated parameters at time t_n are used to simulate the realized failure moments and degradation's for all components. In this way, the simulation models include the updated information. In our system, we apply four updating moments with updating intervals of one year. In the first years after introducing the Collector, there were few Collectors in the field. Little data was collected by these Collectors, which is is the reason for an update interval of one year. We used the cleaned and prepared data as described in Section 3.3 for the Bayesian updating. For the first updating moment, we only used processed data of the first year. The data collected in the second year is the data used for the second updating moment. In this way, we perform the Bayesian updating as close to a real situation with only the available data at that moment.

8.3.3 Simulation implementation

The simulation model that is created in this research is explained in this section. The implementation and results of the complete maintenance model are also presented in this section. The WSM finds the results in this section to highlight the effect of component parameter updating in our model.

Before various maintenance policy models could be verified, their simulation models had to be created first. Different program languages could be used to build a simulation, such as R, Matlab, or Python. In this research, we selected Python as a programming language since it is used within Lely and it is a freely available language. The verification of the simulation model for CBM and ABM components is included in Appendix B. The verifications are performed at the component level with input parameters of Zhu (2015) and Hoedemakers (2020). Despite minor differences between the modelling of ABM and CBM components in our model and the model of Zhu, differences in the verification results can also be explained by the modelling differences.

The system in this study is decomposed to a component level. To find good system solutions, good component solutions should be found first. A grid search approach is applied to find the best maintenance thresholds for both CBM and ABM components in the simulation model. The grid size steps of 1 are used as control limits for the CBM component. For the ABM components, grid steps of 0.1 years in age limit are used. The Pareto optimal component solutions can be found with the component thresholds that give minimal component costs and downtime solutions. These Pareto optimal component solutions are combined with a GA to find the system solutions.

The individual components simulations in this model have realized failure moments or degradation as input. Every set of realized failure moments or degradation is used in simulations with all scenarios of different decision variables for a component once. A combination of sets with realized failure moments and degradation of all system components is called a trace. The number of traces in the system represents the number of simulations for all components in the system. Another input variable of the simulation model is the time horizon T. The time horizon T is set to 15 years in this study since this is the expected lifetime of a Collector. On average, one simulation trace with a time horizon of $T = 15$ takes about 12 seconds of simulation. The duration of the total simulation duration is strongly dependent on the number of traces.

A simulation study with Component D is performed to find a sufficient number of simulations for individual components. In this simulation study, we performed 10 independent simulations with 500 sets of realized degradation moments. The minimal average yearly downtime and cost of these 10 independent simulations are determined to find the standard deviation of the 10 independent runs. The average minimal costs of these 10 independent simulations is 291.9928, with a standard deviation of 1.6805. The average minimal downtime solution has an average of 1.5980 with a standard deviation of 0.016. The standard deviation of the downtime solution is slightly more than 1% of the minimal average downtime solution and a fraction more than 0.5% of the minimal average costs solution. These low percentages are acceptable. Therefore, the simulations in this research are performed with 500 traces.

The system results of a simulation, including the four updating moments of failure and degradation parameters, are shown in Figure 8.3. The WSM is used to find the system results as the first solution approach. This method gives a clear overview of the differences between a model with and without updating. Every single '+' in figure 8.3 represents a system solution found by the WSM with component solutions minimized on costs and downtime and with weights of {0, 1}. A more detailed explanation of the WSM is given in section 6.3. This figure presents all combinations with minimal downtime or cost solutions of components for all maintenance intervals. Since we have 17 (4 CBM and 13 ABM) components with an optimized maintenance threshold, there are $2^{17} = 131,072$ system solutions in the figure per maintenance interval.

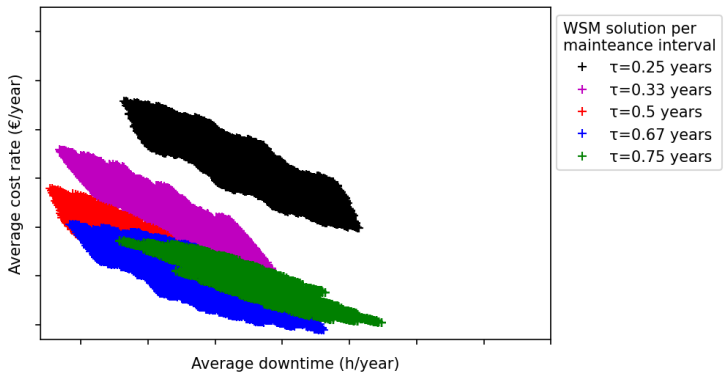

Figure 8.3: System solution with parameter updating found with WSM

To compare the influences of the parameter updating for the system, a simulation without updating is performed. This simulation used the initial parameters, $\eta_{i,0}, \gamma_{i,0}, \alpha_{i,0}$ and β_i to model the degradation of CBM and the failure moments of ABM and FBM component with a Gamma process and Weibull distribution, respectively. The WSM is used to find the system solutions in a model without parameter updating. These system results are shown in Figure 8.4. The difference between Figure 8.4 and Hoedemakers (2020) results can be explained by the slightly different parameters used in the model, the time horizon and the use of the Gamma process instead of the RCM.

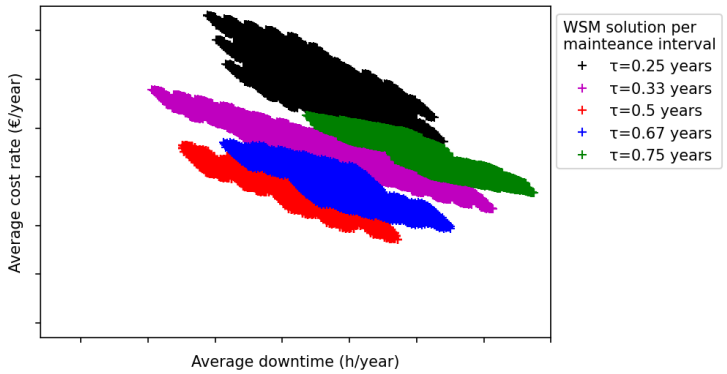

Figure 8.4: System solution with initial parameters found with WSM

Comparing the system results with parameters updating in Figure 8.3 with the results without updating in Figure 8.4, we see that the updating leads to lower average systems downtime and costs. The lower cost and downtime of the system are a result of component parameters updating. The initial component parameters underestimated the lifespan of a component, resulting in too early preventive replacements of components which caused higher system downtime and costs. In addition, it can also be noted that there is a shift in the system solutions per maintenance interval. In the system results without updating in Figure 8.4, a maintenance interval of 0.33 years gives the lowest system downtime, whereas the minimal system downtime solution with updating is at an interval of 0.5 years. The updated parameters can explain this difference. When components fail with another frequency, the ideal maintenance threshold with lower downtime can be reached when maintenance is performed in a different interval.

In the simulation model, there is one optimal system solution in every maintenance interval τ for the minimum system downtime and one for minimum maintenance costs. The system solution that results in the minimum downtime of a machine is denoted by $V^*_{syst}(\tau)$. The system solution for the minimal costs is denoted by $Z^*_{syst}(\tau)$. In Figures 8.3 and 8.4, the minimum downtime solution is in the most left point of the solutions for a particular maintenance interval. The solution that represents the minimum costs is the lowest located solution point of a given maintenance interval.

Table 8.5: Comparison model with and without parameter updating

Maintenance interval	**Optimal downtime**		**Optimal cost**	
τ (years)	$V^*_{syst}(\tau)$ (h/year)	$Z_{syst}(\tau)$ (€/year)	$V_{syst}(\tau)$ (h/year)	$Z^*_{syst}(\tau)$ (€/year)
0.25	5.19%	9.48%	5.08%	8.15%
0.33	5.98%	7.18%	11.51%	9.12%
0.50	8.42%	5.46%	7.86%	10.91%
0.67	9.45%	10.51%	6.84%	14.70%
0.75	10.96%	15.63%	7.86%	17.91%

Table 8.5 shows the relative differences between the considered maintenance intervals of a system with and without updating. The percentages in this Table show that a system without updated parameters results in higher costs and downtime than a system with updating. The updated component parameters are the only difference between the two models. Therefore, the differences in the optimal downtime and cost solution per maintenance interval arise from updating of the parameters. In our situation, the costs and downtime of the system are overestimated since a system with updated parameters results in lower costs and downtime than a system without updated

parameters. The updating of parameters could also result in higher systems costs and downtime. In this case, the initial system underestimates the system costs and downtime since a system with updated parameters results in higher costs and downtime.

The main goal is to create a model as close to the real situation as possible with updated parameters based on new data. A solution with updated parameters should better represents the real situation than non-updated parameters since the parameters are updated based on newly available data. For this reason, we will only consider the system, including the updating of parameters, since this should give results that are closer to a real situation.

8.4 System results including updating

In this section, the system results, including the updated parameters, are presented and analyzed. A GA is implemented to find a better Pareto optimal system solution than found by WSM in section 8.3.3. As concluded from section 8.3.3, the system results, including updating component parameters, should give the results closest to the real situation. Therefore, all results present from this moment include the updating of component parameters unless stated otherwise.

The multi-objective optimization starts at the component level, as explained in section 6.1. The maintenance thresholds of individual CBM and ABM components are first optimized on downtime and costs simultaneously. The optimization of both objectives simultaneously results in a Pareto frontier with component solutions, which is a set of non-dominated component solutions. In section 8.3.3, the WSM was implemented to find system solutions based on combinations of component solutions optimized with respect to either one of the two objectives. For all CBM and ABM components, there are more solutions than the minimal costs or downtime solution. Table 8.6 shows components' maintenance thresholds that give the minimal costs solution or the minimal downtime solution.

Table 8.6: optimized component solutions with $\tau = 0.67$ years

Component name	**Type of maintenance**	**Maintenance threshold** optimized on		**Solutions in Pareto front**
		Downtime	Costs	(p_i)
Component A	CBM	93	77	17
Component B	CBM	22	5	18
Component C	CBM	62	61	2
Component D	CBM	62	57	6
Component E (2)	ABM	0.7	0.1	7
Component F (2)	ABM	4.9	3.9	11
Component G (2)	ABM	7.4	3.9	36
Component H	ABM	2.6	3.3	8
Component I	ABM	0.7	1.4	8
Component J	ABM	0.7	1.4	8
Component K	ABM	0.7	1.4	8
Component L (2)	ABM	3.3	7.6	44
Component M	ABM	1.8	4.9	32

Looking at Table 8.6, we see the control limits of CBM components that provide either the minimal downtime or minimal cost solution for a CBM component. For ABM components, we see the age limits for the minimal costs or downtime solution. Additionally, we see the number of component solutions of which the component Pareto front, p_i, consists. The solutions of these Pareto fronts consist of all possible solutions from the grid search approach between the control limits that gave

the minimal downtime or minimal cost solution. For Component A, 17 Pareto optimal solutions could be used for a system solution. The same applies to all other components. The different system solutions can be created by the product of the number of solutions in the Pareto fronts of all components. The number of possible system solutions is given by: $\prod_{i \in I_{ABM} \cup I_{CBM}} p_i$. In this study, the number of possible system solutions is approximately 7.16×10^{18}. We assume that duplicate components are maintained with the same policy, allowing the duplicate components to be included once. Since our system consists of components that appear twice, such as Component E, F, G and L, the number of system solutions can be reduced to 5.87×10^{13}.

A GA, as described in section 6.5, is implemented to find the best system solutions. The implemented GA in this research consists of 500 generations with a population size of 500 solutions. The fitness of solutions is determined with the NSGA-II approach. In the selection phase of the GA, the best 250 solutions are selected together with 50 randomly selected solutions from the tournament selection with a tournament size of 2. These 300 solutions are used to generate 200 new solutions with a two-point crossover. For these 200 new solutions, there is a change of 10% for a random reset mutation. All solutions are still feasible after a random reset mutation, which is not necessarily the case for a swap mutation. The duration of finding the GA solutions for one maintenance interval took about 100 seconds.

The implementation results of a GA to find Pareto optimal system solutions is shown in Figure 8.5. The dots '.' in this figure show the system solutions that the GA finds. The solutions found by the WSM are also presented in this figure. Every single dot means similar to the '+', a possible system solution. As can be seen from Figure 8.5, most solutions found by the GA present a line that is lower located than the WSM solutions. Since we have to deal with two minimization objectives in our objective function, we can conclude that the solutions found by the GA present better solutions than those presented by the WSM. We also see in Figure 8.5 that a maintenance interval of 0.5 gives the system solutions with the least maintenance downtime. A maintenance interval of 0.67 years gives the system solutions with the lowest costs.

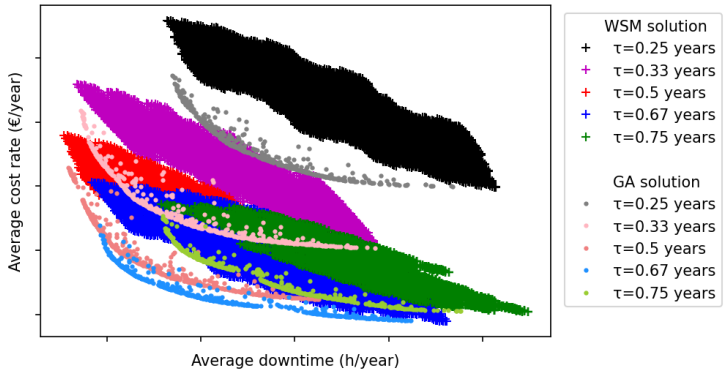

Figure 8.5: System solutions found by a GA

When we zoom in to the results of the GA from Figure 8.5, we see that the GA does not find the system solutions for minimal downtime and costs. The figure shows a line of light blue dots with solutions found by the GA that do not reach the minimal system downtime and cost solutions. The minimal cost solution for a maintenance interval of 0.67 is the lowest point of the dark blue WSM solutions. The minimal downtime solution for this maintenance interval is the most left point of the dark blue solutions. To extend the line with GA solution to the optimal solutions, we used the Pareto frontier of the WSM as input for the initial population of the GA.

In Figure 8.6, we see the zoomed-in system results with a maintenance interval of 0.67 years. In this figure, we can see more clearly that the solutions found by the GA do not find the minimal

system downtime and costs solution of the WSM. The Pareto frontier of the WSM is also shown in this figure by black dots. These black dots show all Pareto optimal solutions of the WSM. When we use these Pareto optimal solutions as partial input for our GA, we get the results showed in Figure 8.7. The same system results found with the WSM and its Pareto frontier are also plotted in Figure 8.7.

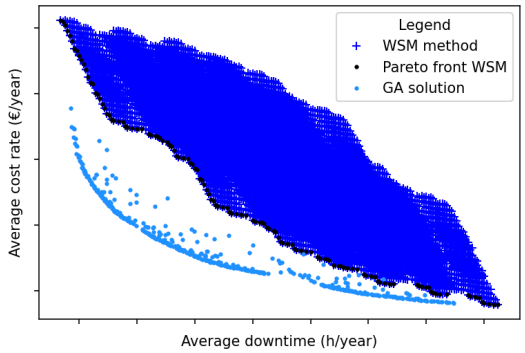

Figure 8.6: System solutions found by a GA with a random initial population with τ of0.67 years

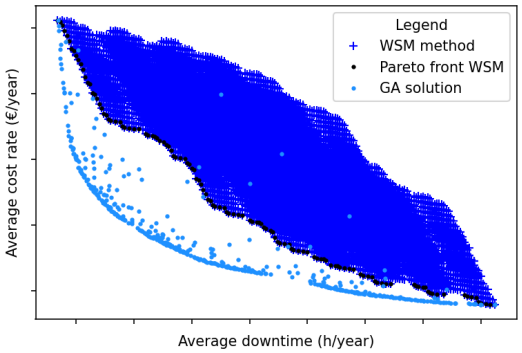

Figure 8.7: System solutions found by the GA with WSM Pareto solutions in the initial population and τ of 0.67 years

Figure 8.6 and Figure 8.7 show the difference between the GA results with a random initial population and a GA with the WSM Pareto optimal solutions in an initial population, respectively. We can conclude from these figures that a GA with the WSM Pareto optimal solutions in the initial population provides better results. The light blue dots representing the GA solutions in both figures are more widely distributed in Figure 8.7 and lie on the minimal costs and downtime solutions. This is an expected result when you start a GA with good initial solutions. Most of the light blue dots together form a convex line in Figure 8.6 and Figure 8.7. These formed convex lines of light blue dots represent the Pareto frontier of the system at a maintenance interval of 0.67. The Pareto frontier of Figure 8.7 shows a more complete Pareto frontier which also includes the minimal system downtime and cost solutions.

We implemented a GA with the WSM Pareto optimal solutions in the initial population for all maintenance intervals. This gave us the most complete Pareto frontiers for each maintenance interval. We looked at these Pareto frontiers to find the Pareto frontier at a system level. This Pareto frontier is found by a combination of Pareto optimal solutions from maintenance intervals of 0.5 and 0.67 years which was also observed in Figure 8.5. The Pareto frontier of the system solutions is shown in Figure 8.8. The different colours of the dots in this figure represent the different maintenance intervals. Every single point of this Pareto frontier represents a Pareto optimal system solution that can be selected to maintain the system. Any advice is given on choosing solutions from the Pareto frontier that can be applied in a practical setting in Appendix C.

Figure 8.8 also includes the current maintenance schedule. The solution of the current maintenance schedule is found with initial parameters as input for the model without updating and with the maintenance thresholds available from the preventive maintenance schedules. In the current situation, the preventive maintenance visits occur with an interval of 6 months ($\tau = 0.5$). Some components are maintained with a different maintenance policy in the current situation than in our model. This is because we started with the advised maintenance policies of Hoedemakers (2020) research instead of the used policies in the current preventive maintenance schedules. There is one adjustment made by modelling the current situation, which is for Component E. We assumed a stochastic dependency for Component E. When Component E fails, an additional component also

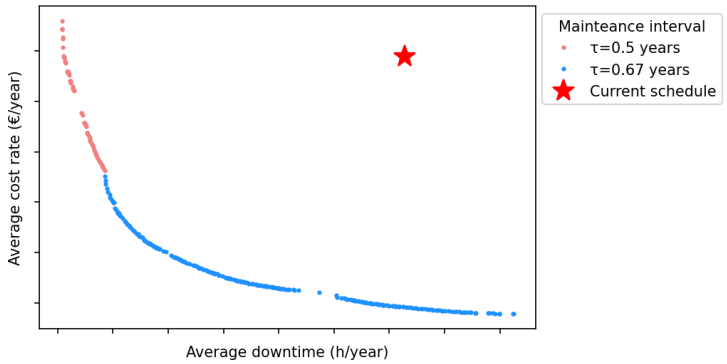

Figure 8.8: Pareto optimal system solutions found by a GA with WSM Pareto optimal solutions in the initial population

needs to be replaced. The solution Component E of the current situation is approximated by the average of the minimal downtime and minimal cost solution of Component E with a maintenance interval of 0.5 years and without parameter updating.

8.5 Component updating results

In this section, we look at the component results of the simulation with a maintenance interval τ of 0.67. This maintenance interval is selected because it generally provides the best system results. The simulation results of the CBM components with a τ of 0.67 are presented in Table 8.7. The second to the fourth column of this table shows the component results for the minimal components downtime solution. The second column shows the condition threshold C_i^{V*} that gives the minimal downtime solution. The downtime and costs associated with this control limit for minimal downtime are presented in the third and fourth column. The downtime and costs solution with a control limit that minimized the downtime are denoted by $V_i(C_i^{V*})$ and $Z_i(C_i^{V*})$, respectively. The fifth until the seventh column shows the component solution with minimal component costs where C_i^{Z*} is the control limit that gives the minimal cost solution.

Table 8.7: Optimal CBM component solution with $\tau = 0.67$

	optimized on downtime			**optimized on costs**		
Component name	C_i^{V*}	$V_i(C_i^{V*})$ (h/yr)	$Z_i(C_i^{V*})$ (€/yr)	C_i^{Z*}	$V_i(C_i^{Z*})$ (h/yr)	$Z_i(C_i^{Z*})$ (€/yr)
Component A	93	0.46	122.34	77	0.55	104.37
Component B	22	0.49	196.24	5	0.63	154.12
Component C	62	0.63	75.71	61	0.63	75.35
Component D	62	0.47	99.20	57	0.54	90.64

The control limits presented in Table 8.7 should be interpreted in the following manner. When the component condition is below a control limit C_i, the component must be replaced during a PM visit. To maintain Component A with minimal impact on the downtime of the Collector, Component A should be replaced during every PM when the condition of Component A is below 93. Performing maintenance for Component A with a control limit of 93 gives on average 0.46 hours of system downtime per year an average cost of €122.34. To find the control limit that results in the minimal maintenance costs for Component A, a control limit of 77 should be used. This results in an average yearly downtime of 0.55 hours and an average yearly cost of €104.37. The solutions of the other CBM components can be interpreted similarly.

For all CBM components, the condition threshold that results in minimal downtime is higher than the condition threshold that results in the minimal cost solution ($C_i^{V*} > C_i^{Z*}$). This indicates that a higher condition threshold results in less downtime and higher costs. The condition of all CBM components decreases over time. Therefore, a higher condition threshold indicates that components are replaced earlier. Understandably, the control limits for minimal system downtime are higher than the control limits for minimal costs since the ratio between CM and PM downtime is larger than the ratio of costs, i.e., $d_i^{CM}/d_i^{PM} > c_i^{CM}/c_i^{PM}$.

Now that we understand how to read component outcomes, we can investigate how the component parameter replacements influence these outcomes. Figure 8.9 shows the effect of parameter replacements on the component solution per updating moment. A component simulation is performed with the initial parameters to find the component solution for zero update moments. Two updating moments stand for two moments of parameter replacements, one after the first year and one after the second year. In this example, with two parameter replacements in the first and second year and a time horizon of $T = 15$, all years after the second update moment are simulated with the parameters found by the second update moment. Therefore, the remaining 13 years in the time horizon are simulated based on the parameters of update moment two in this example. This simulation approach is performed to find the results shown in Figure 8.9 and Figure 8.10. Figure 8.9 shows the minimal downtime solutions with the associated control thresholds of Component B. The number of update moments indicates the frequency of parameter replacements with a time interval of one year after the start of the model. A simulation with one update moment finds the result with initial parameters in the first simulation year and the updated parameters for the remaining 14 simulation years.

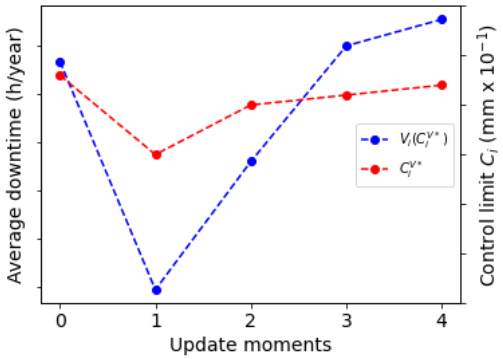

Figure 8.9: Parameter updating effect of Component B on the control limit and average downtime with a τ of 0.67 years

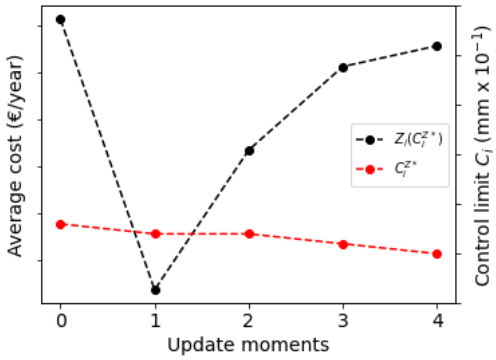

Figure 8.10: Parameter updating effect of Component B on the control limit and average costs with a τ of 0.67 years

In Figure 8.9, we see the optimal control limit presented by the red line, which is more or less the same for all updating moments. The average downtime is presented by the blue line which, does not remain constant. We see that the average downtime is much lower at the first updating moment than for all other results. The same pattern is shown in Figure 8.10 for the costs presented with the black line. In this figure, the control threshold is almost constant over all updating moments. The maintenance interval can explain the almost constant control thresholds for both average costs and downtime. The maintenance interval is constant and the changes in degradation parameters are limited. Therefore, the control limit remains more or less constant at all updating moments. The decrease in downtime and cost can be explained by the parameters used during the first updating moment. The initial parameters showed results with an overestimated degradation. Besides that, the updated parameters resulted in lower degradation of Component B, resulting in less frequent PM and CM replacements of tComponent B. Less frequent replacements result in

the end in lower costs and downtime of a component.

Table 8.8 shows the component results of the ABM components with a maintenance interval of 0.67 years. This table is similar to the table with the CBM results except for the control limit C_i that is changed to the age limit A_i. To find the component with minimal downtime, the age-limit A_i^{V*} should be used. Age limit A_i^{C*} denotes the age limit that results in the minimal cost solution for a component i. The average yearly downtime and average yearly costs are represented by $V_i()$ and $Z_i()$ respectively.

Table 8.8: Optimal ABM component solution with $\tau = 0.67$

	optimized on downtime			**optimized on costs**		
Component name	A_i^{V*} (yrs)	$V_i(A_i^{V*})$ (h/yr)	$Z_i(A_i^{V*})$ (€/yr)	A_i^{Z*} (yrs)	$V_i(A_i^{Z*})$ (h/yr)	$Z_i(A_i^{Z*})$ (€/yr)
Component E (2)	0.7	0.14	42.54	0.1	0.22	37.43
Component F (2)	4.9	0.04	14.93	3.9	0.05	12.27
Component G (2)	7.4	0.06	24.81	3.9	0.08	10.75
Component H	2.6	0.08	11.57	3.3	0.08	10.91
Component I	0.7	0.04	18.19	1.4	0.06	14.76
Component J	0.7	0.05	7.00	1.4	0.06	6.76
Component K	0.7	0.06	7.44	1.4	0.06	7.23
Component L (2)	3.3	0.09	23.98	7.6	0.10	16.39
Component M	1.8	0.06	26.26	4.9	0.12	18.98

A component should be preventively replaced during a SD when the age of a component is above the age limit. This preventive replacement should happen in the time interval A_i, $A_i + \tau$. Another possibility in this time frame is that the component fails before the SD. In this case, the component is correctively replaced during a USD. When we look at the age limit for the Wall guide that gives the minimal downtime solution, we see that $A_i = 1.8$ years. With a τ of 0.67 years, a PM can happen between an age of 1.8 and 2.47 years. If the previous replacement happened during a SD and the component does not fail, this component will be replaced with an age of 2.0 years. This is visualized in Figure 8.11. When a CM replacement occurs 0.5 years before a SD, the next preventive replacement of this component will be at the age of 1.84 years if the component does fail in these 1.84 years. The 1.84 years is found by 0.5 years plus two maintenance intervals (1.84=0.5+2×0.67). This scheduling is easily implementable when the age of components in a is traced.

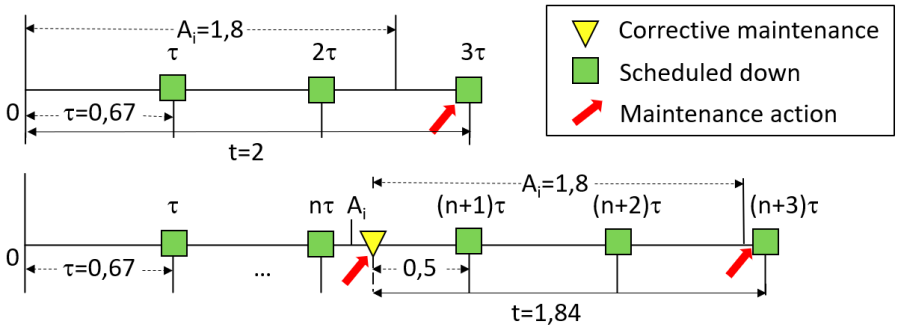

Figure 8.11: Maintenance ABM component with age limit $A_i = 1.8$ and $\tau = 0.67$ years

We observe from Table 8.8 that component age limits that minimize downtime for most components are smaller than the age limits that minimize the costs. For all components, except Component E, F and G, it holds that $A_i^{V*} \leq A_i^{Z*}$. For these components, an earlier preventive

replacement (lower A_i) results in less downtime and higher costs. For Component E, F and G, the opposite is true. The same reasoning as at the CBM components can be used for ABM components. When the ratio between CM and PM downtime is larger than the ratio of costs, i.e., $d_i^{CM}/d_i^{PM} > c_i^{CM}/c_i^{PM}$, a lower age limit results in a minimal downtime solution than the age limit for minimal costs solution. For Component E, F and G, the ratio between CM and PM downtime is smaller than the ratio of costs, i.e., $d_i^{CM}/d_i^{PM} < c_i^{CM}/c_i^{PM}$, therefore the opposite holds. A higher cost ratio than the downtime ratio for these components can be explained by the high additional replacements costs of an additional component during a CM.

The effects of the parameter updating on the minimal downtime solution of Component J are presented in Figure 8.12. This figure can be interpreted similarly to the updating effect of Component B in Figure 8.9. The optimal downtime solution is presented by the blue line and the corresponding age limit with the red line. We see a decrease in the average yearly downtime over the updating moment. The corresponding age limits have a slight increase. The small increase in the age limit can be explained by the increase of the scale parameter. A higher scale parameter results in less frequent failures and therefore, the age limit for minimal downtime has a small increase. We see the same pattern for the optimal cost solution in Figure 8.13. The age limit for the minimal cost solutions has a larger increase than the age limit for optimal downtime. The increasing scale parameter is the reason for this increasing age limit.

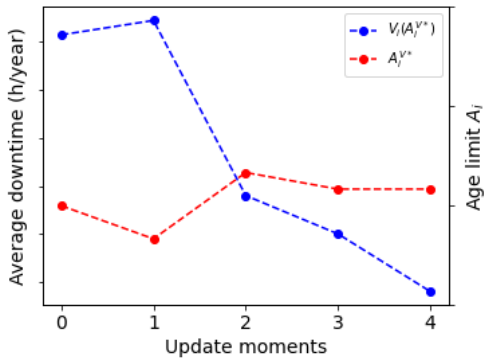

Figure 8.12: Parameter updating effect on the age limit of Component J and average downtime with a τ of 0.67 years

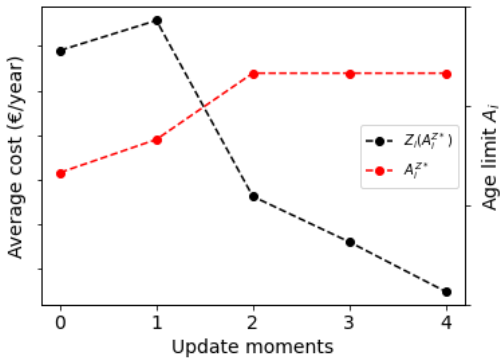

Figure 8.13: Parameter updating effect on the age limit of Component J and average costs with a τ of 0.67 years

The steep decrease in average yearly downtime and cost can be explained by the scale updated parameters. For ABM components, the scale parameters of the Weibull distribution are updated at these updating moments. The scale parameter of the Weibull distribution influences the moment when a failure can occur. In our situation, updating the scale parameter resulted in higher scale parameters for Component J (see Table 8.3). This higher value results in more time between the failure of a component, so less frequent failures of a component. These less frequent failures result in lower average yearly downtime and costs for Component J.

8.6 Sensitivity analysis

In this section, several sensitivity analyses are performed to find the impact on component or system solutions of changes in the input parameters. The sensitivity analyses are performed to show how sensitive the model reacts to a variation of input parameters. These sensitivity analyses are performed on parameters for which there is uncertainty in the model. By performing a sensitivity analysis, we can study how the solution changes due to changes in the value of an uncertain parameter. All analyses are performed on a system, including updating of parameters, similar

to the models' input in the previous section. When other parameters are used in the sensitivity analysis, this will be explicitly mentioned.

Three sensitivity analyses are performed in the following subsections. There is uncertainty about the initial scale parameter used in our model. Therefore, the first analysis is performed to find the effect of the initial scale parameter on the Bayesian updating results in section 8.6.1. There is also uncertainty on the time horizon of a new introduced system. The sensitivity of the time horizon on CBM and ABM components is investigated in section 8.6.2. In section 8.6.3, the system sensitivity on setup downtime and costs for SD and USD is performed. This sensitivity analysis is performed since the distance between Lely Centers and farmers can be different in various regions. A different distance between Lely Center and the farmer could result in different setup downtime and costs in different regions.

8.6.1 Bayesian updating sensitivity

In this section, we investigate the effect of the initial scale parameter of the Weibull distribution on the Bayesian updating results. The initial scale parameters has values of −50%, +50%, +100% and +500% of the true initial scale parameter. It is expected that an initial guessed parameters will normally be within an error range of plus or minus 50%. However, sometimes an initial parameter can be determined completely wrong. Therefore, the effect of the initial scale parameters of plus 100% and plus 500% are also investigated. The updating effects are investigated with four updating moments similar to those as applied in the model in the previous section.

Figure 8.14 and Figure 8.15 show the sensitivity results of Component I and Component E respectively. The green dots on the green line represents the Bayesian updating results used in the model. This initial parameter is determined by a Uniform distribution of one plus and minus the scale parameter used in the research of Hoedemakers (2020). The other dashed lines and dots present the updating results with a certain percentage deviation from the initial green scale parameter. There can be concluded from both figures that the updated scale parameter value gets closer to the green line when more updates are performed. After the second update moment, all updated scale parameters are close, except for the +500% initial scale parameter. The updated value of the +500% initial scale also gets closer to the green line, but it requires more updating moments. After the fourth updating moment, the updated value also reaches the same value as the other updated scale parameters.

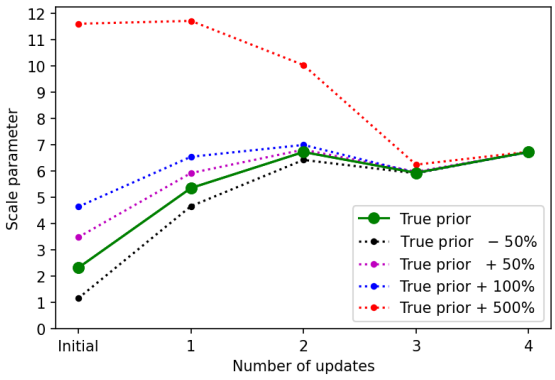

Figure 8.14: Effect of Bayesian updating with different initial scale parameters of Component I

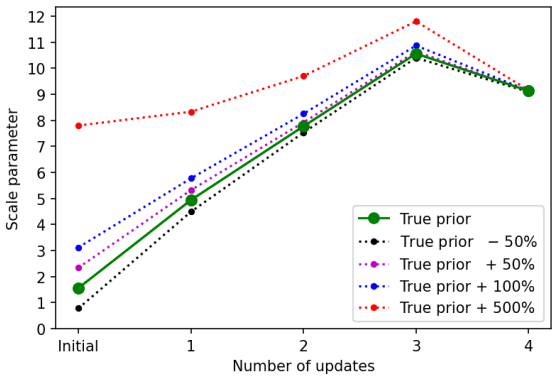

Figure 8.15: Effect of Bayesian updating with different initial scale parameters of Component E

In Figure 8.16 and Figure 8.17, we see the updating results of Component M and K. These components' initial scale parameters are larger those of Component E and I. This results in a larger

spread of the initial scale parameters that are a certain percentage greater or smaller than the initial scale parameter. As a result of this larger spread, we see that the scale parameter +500% does not reach the same value as the green line. However, we see that the updated values of +500% scale parameter are getting closer to the green dots when the number of updates increases. Therefore, we can conclude that updating of scale the parameters with different initial values results in scale parameters that are mostly similar to each other. When an initial scale parameter has a starting value not close to the true initial value, it still reaches the correct updated value (see Figure 8.14 and Figure 8.15). When the gap between the initial parameter and the true initial scale parameter is too big, the updated scale parameter is getting closer to the true updated value. Still, it does not reach the actual value with four updates. More updating moments are needed for the +500% scale to reach the actual value.

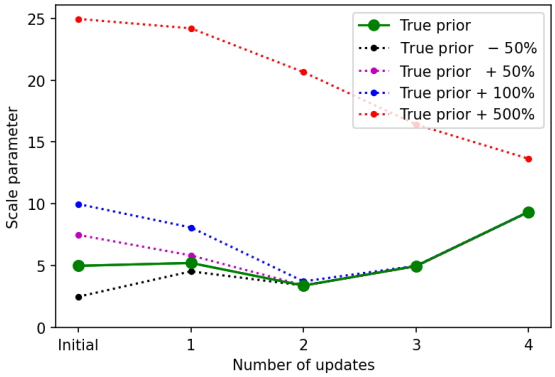

Figure 8.16: Effect of Bayesian updating with different initial scale parameters of Component M

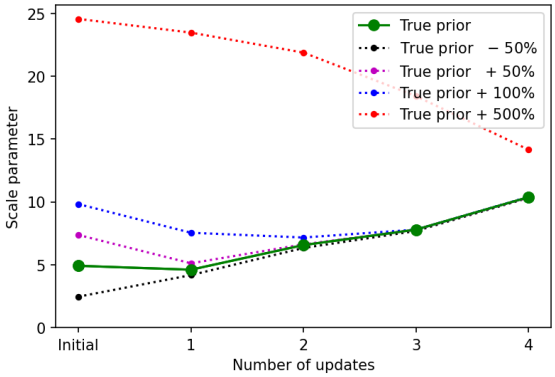

Figure 8.17: Effect of Bayesian updating with different initial scale parameters of Component K

8.6.2 Time horizon sensitivity for components

This section considers the effect of a finite time horizon length for ABM and CBM components. The effects of the time horizon are investigated for both objectives with different maintenance intervals. The associated maintenance threshold that gives the minimal downtime or cost solution in a certain time horizon is also investigated. The minimal component solutions for downtime and costs are considered. Component B and M are selected for these sensitivity analyses for CBM and ABM components, respectively. These components are selected based on the spread between the optimal downtime and cost solution and their updated Weibull or replaced Gamma process parameters. This analysis shows the effect of a component when the time horizon is uncertain. The time horizon of a component is dependent on the expected lifetime of a system. For new systems, it is hard to determine their expected lifetime. Therefore, this analysis is performed.

8.6.2.1 Condition-based components

The results of the sensitivity analysis of Component B are investigated in this section. Figure 8.18 shows the minimal downtime solution for Component B with different maintenance intervals, including the replaced parameters. The horizontal axis presents the length of the finite time horizon T. In this figure, we see a clear order in maintenance interval with their associated downtime. A maintenance interval of 0.25 years gives the lowest downtime, while a maintenance interval of 0.75 years results in the highest downtime. Besides that, we also see a dip in the average yearly downtime of all maintenance intervals in the second year. At the start of the second year, the initial degradation parameter is replaced with new degradation parameters that provide lower degradation. A component with less degradation should be less frequently preventive or corrective

replaced, which results in less downtime. Therefore, there are lower average costs and downtime in the second year. All other replacements of degradation parameters resulted in a higher degradation whereby the average downtime increases after the dip. Furthermore, we see that the average yearly downtime of all maintenance intervals increases towards a horizontal asymptote. This horizontal asymptote is the infinite solution of maintaining this component with the last replaced parameters ($\gamma_{i,4}$ and $\eta_{i,4}$) and the same maintenance interval. The finite horizon solution approaches the infinite horizon solution when the time horizon increases.

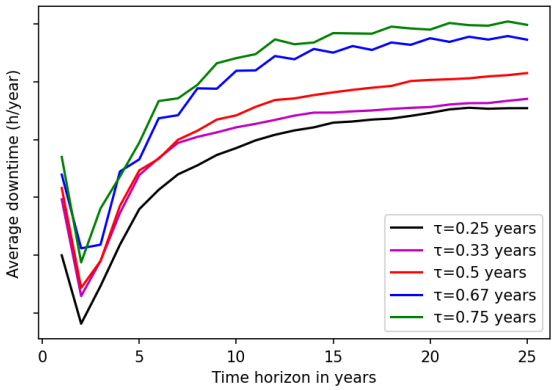

Figure 8.18: Average yearly downtime of Component B in a time horizon of 25 years

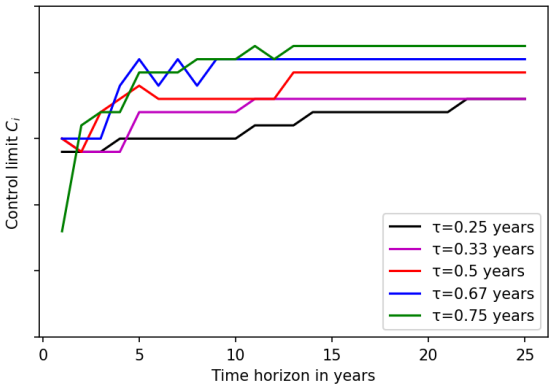

Figure 8.19: Optimal control limit of Component B with minimized downtime in a time horizon of 25 years

In Figure 8.19, we see the control limit of the different maintenance intervals associated with the minimal downtime solution of Component B. We see the same order of the lines that present the different maintenance intervals as in Figure 8.18. Lower maintenance intervals have lower control limits, resulting in the lowest downtime for Component B. In Figure 8.19 we also see a small increase in the optimal control limits for all maintenance intervals. The jerky changes in the control limits can be explained by the step size used as possible control limits. This increase leads to the optimal control limit with minimal downtime for maintaining Component B with a given maintenance interval.

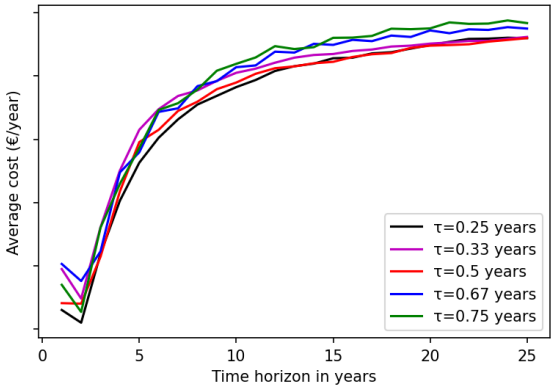

Figure 8.20: The average yearly cost of Component B in a time horizon of 25 years

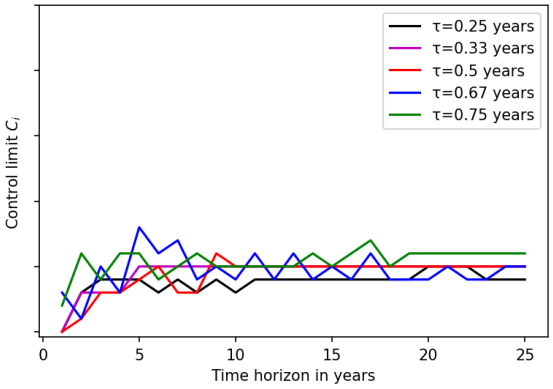

Figure 8.21: Optimal control limit of Component B with minimizing costs in a time horizon of 25 years

Figure 8.20 and 8.21 show the minimal average yearly costs and the associated control limits for these costs at different maintenance intervals. In Figure 8.20, we see the same dip in the second

year as in Figure 8.18. This dip can also be explained by a decrease in degradation from the replaced parameters. The same pattern of a steep increase followed by a flattening line as in Figure 8.18 is shown in Figure 8.20. The same reason for the downtime solution holds for the cost solution.

In Figure 8.25, we see a small increase in maintenance thresholds that gives the optimal costs solution. This is a smaller decrease as in the optimal downtime solution. The smaller increase of the control limit than the downtime solution can be explained by comparing the ratio between cost and downtime associated with CM and PM activities. The CM, PM ratio for downtime is larger than for costs, $d_i^{CM}/d_i^{PM} > c_i^{CM}/c_i^{PM}$. Therefore, relatively more downtime can be prevented by increasing the control limit for the downtime solution than for a cost solution. A higher control limit leads to downtime of earlier preventive replacements, but it also prevents higher downtime caused by corrective replacements. For the costs, the same can be applied. A lower ratio results in less increase in control limit over time than downtime.

8.6.2.2 Age-based components

This section investigates the effect of the time horizon length on the optimal costs and downtime solutions for an ABM component. Figure 8.22 shows the average downtime of Component M with different maintenance intervals over a time horizon of 25 years. For all maintenance intervals, the same pattern of first a significant increase in average downtime followed by a slighter decrease after reaching the peak around year seven is shown in Figure 8.22. This significant increase in the first years can be explained by the decrease in the updated scale parameters in the first and second update moment. This results in earlier failures and therefore more downtime. After the third and fourth update moment, the updated scale parameter increases. This results in less frequent failures and therefore less downtime. The peak in the seventh year can be explained due to the simulated failures occurring later than the moment of updating. The same pattern of reaching a horizontal asymptote as at a CBM component can be seen from year 10. This asymptote is approached from above due to the higher costs in the first years.

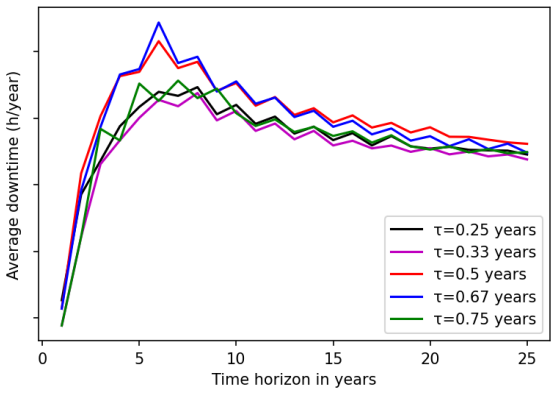

Figure 8.22: The average yearly downtime of Component M in a time horizon of 25 years

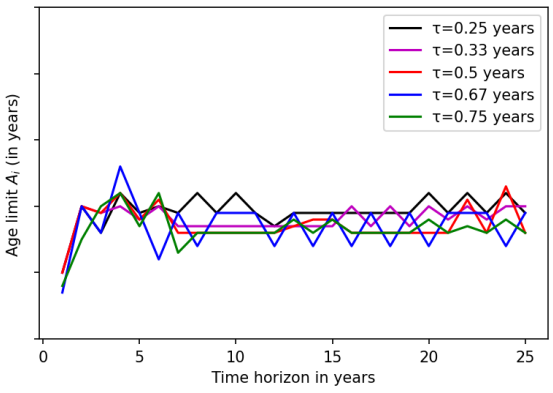

Figure 8.23: Optimal age limit of Component M for minimizing downtime in 25 years time horizon

The age limits for optimal downtime stay more or less constant over the time horizon T. Figure 8.23 shows only a small increase of the age limits in all maintenance intervals until year 4. The updated parameters cause this increase. After year 4, the age limits stay more or less constant. This low constant age limit implies that it is better to prevent a failure than let failures occur at a higher age limit. This can be explained since the setup downtime for a CM is higher than for a PM.

Figure 8.24 shows the same pattern for costs as the downtime has in Figure 8.22. The same explanation of the updated parameters can be applied to explain the pattern in this figure. Figure

8.25 shows a large increase of all age limits associated with the optimal cost solutions for all maintenance intervals. This increase is not a smooth increase of the optimal age limits. The large increase in the age limits can mainly explain the increase in updated parameters. Since the ratio of setup costs is smaller than for setup downtime, we see an increase in the age limit over the time horizon, which we only see minimally in Figure 8.23.

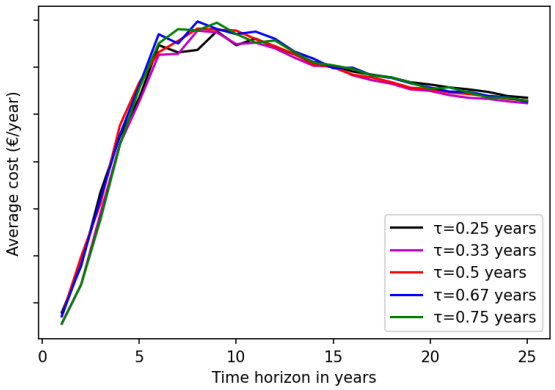

Figure 8.24: The average yearly cost of Component M in a time horizon of 25 years

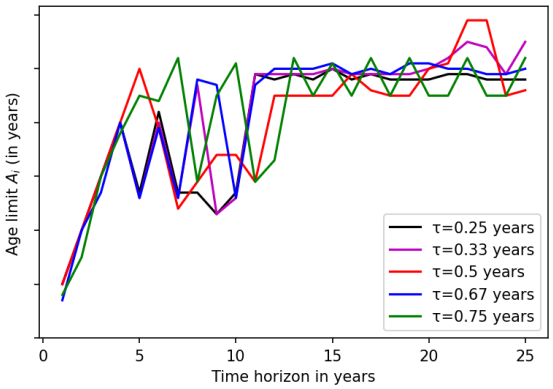

Figure 8.25: Optimal age limit of Component M for minimizing costs in 25 years time horizon

8.6.3 System sensitivity on setup costs and downtime

This section investigates the sensitivity of the setup costs and downtime of SD and USD maintenance activities. This is interesting to investigate since the distance between Lely Centers and the farmers can be different in various countries or regions. The associated setup costs and downtime for scheduled and unscheduled system downs might also be different in different regions.

To investigate the system's sensitivity on setup costs and downtime, we performed simulations with 50% higher and 50% lower setup downtime or costs. For each simulation, we only adapted the downtime or costs for either the SD or the USD. This results in eight different scenarios that we simulated and compared to the normal situation. The results of this sensitivity analysis with a maintenance interval of 0.67 years are shown in Table 8.9. The first two columns of this table show the changes in setup downtime and costs compared to the normal situation. The third and fourth columns show the system solutions in downtime and costs when the system is minimized on downtime. The fifth and sixth columns show the system solutions when the system is minimized on costs.

Table 8.9: Effects of setup downtime or costs on the system solution.

		Optimal downtime		**Optimal costs**	
		$V_{syst}(\tau)$	$Z_{syst}(\tau)$	$V_{syst}(\tau)$	$Z_{syst}(\tau)$
d^{S-USD}	+50%	24.73%	-0.43%	29.82%	-1.15%
d^{S-USD}	−50%	-29.12%	-1.53%	-30.68%	-0.64%
d^{S-SD}	+50%	5.04%	-6.02%	5.00%	-0.51%
d^{S-SD}	−50%	-5.13%	-4.87%	-4.14%	-0.66%
c^{S-USD}	+50%	-0.27%	8.26%	-5.00%	13.31%
c^{S-USD}	−50%	0.09%	-12.81%	6.79%	-15.58%
c^{S-SD}	+50%	-0.37%	3.62%	1.17%	6.21%
c^{S-SD}	−50%	0.18%	-4.76%	0.31%	-7.02%

In Table 8.9, we see that the setup downtime of an USD has the most influence on the system solutions. We see that an increase of 50% setup downtime does not lead to an increase in downtime of 50%. However, an increase of 50% setup downtime leads to almost 25% more downtime when minimizing the system on downtime and an increase of almost 30% more system downtime when minimizing the system on costs. Besides that, we see that USD cost or downtime changes have the most effect on either the system costs or the system downtime. The initial USD setup downtime and costs are higher than the initial SD setup downtime and costs. Therefore, an increase in USD setup costs and downtime has more effect on the system solutions than SD setup changes.

Chapter 9

Conclusions

In this last chapter, we describe the answers to the research questions posed in this study. Developing an optimal preventive maintenance concept with updating options based on newly available service data is challenging since there is limited available data when a product is introduced to the market. In the research, we had to deal with this data to answer the following main research question:

How to optimize a preventive maintenance concept for products in the early exploitation phase, based on field experience?

This research question is answered by answering the four sub-questions. The answers to these sub-questions together answer the main research question. Section 9.1 gives the answers to the research questions presented in chapter 2. In Section 9.2, the limitations and recommendations for future research are stated. Lastly, the recommendations for Lely are given in Section 9.3.

9.1 Conclusion

In the previous chapters, four research questions were defined and answered. The study is summarized and the answers to the research questions are presented in this section.

1. How to deal with data collection from products for maintenance purposes?
In Chapter 3, this first research question is answered. First, literature about different types of maintenance data is discussed. We found that maintenance data can be distinguished into four categories: Asset history data, Usage data, Stressor data and Condition data. During this research, we focused on the Asset history data and especially on the historical data. This category includes the available service data at Lely. The available service data of Lely contains some shortcomings, which make this data hard to use for maintenance purposes. There is no clear distinction between the types of maintenance is used. Besides that, the underlying reason why a machine stopped working is not properly and uniformly stored data. The literature is consulted on how to correctly and consistently collect service data. The number of possible maintenance categories should be reduced so a service engineer can easily select the correct category. For all maintenance types, general information should be collected to find all maintenance actions of one machine. Besides that, additional information is recommended to collect during the different maintenance types. To make this data easier to use, it is recommended to collect it by selecting one of the different categorical options. For example, it is preferred to know the reason of a machine breakdown. Furthermore, different information about different components is recommended to collect. For CBM components, it is recommended to collect the condition of the components during inspections and replacements. This is helpful information to determine the degradation pattern of a CBM component.

Additional to the recommendations of collecting the service data, we still had to work with the currently available service data. Several steps of data processing and data cleaning are performed to the available service data. As a result of these data processing and cleaning steps, we got usable data during the rest of this research.

2. Which time horizon; finite, rolling or infinite, is best suitable in a multi-component maintenance model for Lely?
The second research question is answered by literature research in Chapter 4. First, the differences between the three different time horizons are investigated after a literature review of multi-component maintenance models is conducted. From this literature research, there is concluded that a maintenance model with a finite horizon is best suitable for the situation at Lely. This horizon allows the use of newly available data in the model. This is one of the model requirements from Lely, updating a maintenance model using newly available service data. Additional to the updating requirement, Lely sometimes replaces a system component with a new updated version of this component. This is only possible in a maintenance model with a rolling or finite horizon. Besides that, a finite horizon might be more appropriate when the expected lifetime of one or more components is close to the intended system lifetime.

In the literature research for a multi-component maintenance model, we looked for a model with a finite horizon. We did not find a suitable multi-component maintenance model with different maintenance policies. Therefore, the maintenance model of Zhu (2015) is used as the basis for this research. This model contains an infinite time horizon, which meant the time horizon had to be adjusted to apply the update requirement of Lely. The model of Zhu (2015) is implemented to find a solution through simulations with a finite horizon. This finite horizon allows updating with the use of newly available data. A suitable updating method applied in a maintenance model is found by Shi et al. (2020). This method is used as starting point for the updating procedure and the implementation in a maintenance model.

3. How to include updating of components failure information based on new evidence in a maintenance concept?
Chapter 7 answers this third research question. The answer to this research question is given in two sections, one for the Weibull parameters of ABM and FBM components and one for the Gamma process parameters of CBM components. The updating procedure for ABM and FBM components is performed by Bayesian updating. The Gamma process parameters are replaced by new parameters found by a distribution fit on the data.

The Bayesian updating procedure for ABM and FBM components is performed on the scale parameter of the Weibull distribution. We assumed that the estimated shape parameters found in the research of Hoedemakers (2020) are correctly estimated with a given uncertainty. This uncertainty is model by a Uniform distribution to find the initial shape parameters. We assume the same Uniform distribution for the initial scale parameters. This initial scale parameter is yearly updated with the newly available service data. A Bayesian updating model is created to update the Weibull scale parameters with the available service data yearly.

We estimated the Gamma process parameters by a distribution fit with the MLE on all available data. Before estimating the Gamma process parameters, the processed and cleaned service data from Chapter 3 needed some further steps to find the degradation rates of a component instead of lifetimes. The same update interval of a year is used for estimating the Gamma process parameters.

4. How can the framework be applied to the Lely Collector?
This research question is answered by a case study on the Collector in Chapter 8. A simulation model including the updating of parameters is created to find solutions to the multi-component model. This model is applied to a case study of the Collector. The model results include updating to give a better presentation of the reality since the updated input parameters are based on

the actual failures of components. The system results with updating show lower average system downtime and costs than the system without updating. This is mainly caused by the underestimation of the component lifetimes. When a component lifetime is underestimated, the component is replaced more frequently in the model, resulting in higher costs and downtime.

The system solutions in this research are found by combinations of all Pareto optimal component solutions. At system level 5.87 $\times 10^{13}$ different combinations with the component solutions are possible. To find the best system solutions out of these options, a Genetic Algorithm (GA) is implemented. With the implementation of a GA, better Pareto optimal system solutions were found than with the Weighted Sum Method (WSM). The Pareto optimal solutions of the WSM used as input for the GA resulted in even better Pareto optimal system solutions found by the GA. Most solutions of GA together results in a line that presents the Pareto optimal system solutions.

9.2 Limitations and future research

The limitations of our research, together with the recommendations for future research, are presented in this section.

A limitation of our research is the updating of only the scale parameter from the Weibull. The failure behaviour of a component modelled by a Weibull distribution consists of a scale and shape parameter. Updating both parameters based on newly available data should give a better presentation of the real situation based on the data. According to the data, the assumed value of the shape parameter could be different from the true shape parameter.

The updating procedure in this study to update the scale parameter of the Weibull distribution was based on a software package in Python. Verification showed that this package provided satisfactory results. However, implementing an updating procedure based on literature should be a more appropriate method for the updating procedure. A recommendation for future research is to find and implement an updating procedure based on a theoretical proven approach.

Another limitation and together recommendation for future research is the consideration of multiple failure distributions. In our research, we assume that the failures of ABM and FBM components are modelled by a Weibull distribution. During the updating procedure with newly available data, we only consider a stationary Weibull distribution. However, it could be that another distributions better models the failure behaviour of a component than the Weibull distribution. Therefore, a recommendation for future research is to consider also other distributions than only the Weibull distribution.

In our research, we only consider positive economical dependencies between different components in our model. In a real situation, there are also stochastic and structural dependencies between components of the Collector. In our research, we tried to include stochastic dependencies by increasing the CM costs of some components. However, there are better methods to model the structural and stochastic dependencies in a maintenance model. For future research, it is recommended to investigate these dependencies and include them in the model.

The condition degradation in our research is modelled with a Gamma process. The parameters for this Gamma process are found by transforming service data into degradation rates. This is not the standard method to find the parameters of a Gamma process. Normally these parameters are estimated with available condition data of components over time. This information was not available during our study, but this information should become available by collecting this data via IoT for future research. Another method to estimate these parameters is by collecting conditions of components during PM inspections. These actions are currently performed during PM. However, the component condition information is not stored yet.

9.3 Recommendations for Lely

Service data collecting

The first recommendation for Lely is to collect their service data uniformly and correctly. For example, error causes are not filled in consistently and uniformly during breakdowns of a machine. When the data is not filled in uniformly and consistently, useful information from machine breakdowns could be missed. Reducing and renaming the possible options to uniformly interpretable error causes makes it easier to select the correct error cause during a breakdown for a service engineer. Furthermore, by making it compulsory to fill in this information after every breakdown, more data on machine error causes will be collected. This should give a better presentation on the major error causes of a system.

Collect system utilization information

Collecting data about system utilization is another recommendation for Lely. Multiple components of the Collector fail due to wear. The wear of these components can be likely linked to the utilization of a system. Since not all farmers use the Collector in the same way with the same utilization, components in a Collector at one farmer can wear faster than at another farmer. The distance driven, the number of dumps or liter of manure collected are indicators of the utilization of a Collector. By finding the relationship between the wear of a component and the utilization of a Collector, more individualized maintenance concepts could be developed based on the utilization of a Collector at a certain farm.

Uncertainty in setup costs and downtime

In the sensitivity analysis, we looked at the influence of changes in setup costs and downtime on the system solution. The distance between a farmer and a Lely Center is not equal in all areas. This sensitivity analysis showed that the increases and decreases of unscheduled setup downtime and costs have the most effects on the system solutions. Therefore, it is recommended to properly determine the different setup downtime and costs for USD in the model for different regions since this has the most effect on the system solutions. For the individualization of a maintenance concept, different setup costs for different farmers or Lely Centers should be considered since this will result in different optimal system solutions.

Predictive maintenance

The model created in this research is focused on Preventive maintenance (PM) of a multi-component system. A follow-up step after PM is predictive maintenance. In this maintenance type, the failures of a component or system are predicted based on current information and historical data. Predictive maintenance is only possible when predictions of failures moments can be made based on the actual condition. This is possible with condition monitoring. However, it is currently not possible to remotely see the (condition) information of a Collector. Therefore, collecting (condition) data of the Collector via IoT is required before the implementation of predictive maintenance is even possible.

References

Arts, J. J. (2017). *Maintenance modeling and optimization*. BETA publicatie : working papers. Technische Universiteit Eindhoven.

Baker, R. and Christer, A. (1994). Review of delay-time or modelling of engineering aspects of maintenance. *European Journal of Operational Research*, 73(3):407–422.

Beck, J. L. and Katafygiotis, L. S. (1998). Updating models and their uncertainties. i: Bayesian statistical framework. *Journal of Engineering Mechanics*, 124(4):455–461.

Boon, M., van der Boor, M., van Leeuwaarden, J., Mathijsen, B., van der Pol, J., and Resing, J. (2020). *Stochastic Simulation using Python*. Department of Mathematics and Computer Science Eindhoven University of Technology.

Cho, D. I. and Parlar, M. (1991). A survey of maintenance models for multi-unit systems. *European Journal of Operational Research*, 51(1):1 – 23.

de Jonge, B. and Scarf, P. A. (2019). A review on maintenance optimization. *European journal of operational research*.

Deb, K. (2014). *Multi-objective Optimization*, pages 403–449. Springer US, Boston, MA.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197.

Dekker, R. and Wildeman, R. (1997). A review of multi-component maintenance models with economic dependence. *Mathematical Methods of Operations Research*, 45(3).

Drent, C., Kapodistria, S., and Boxma, O. (2020). Censored lifetime learning: Optimal bayesian age-replacement policies. *Operations Research Letters*, 48(6):827 – 834.

Ebeling, C. E. (2019). *An Introduction to Reliability and Maintainability Engineering*. Waveland Press.

Geitner, F. K. and Bloch, H. P. (2012). Chapter 9 - the "seven cause category approach" to root-cause failure analysis. In Geitner, F. K. and Bloch, H. P., editors, *Machinery Failure Analysis and Troubleshooting (Fourth Edition)*, pages 615 – 635. Butterworth-Heinemann, Oxford, fourth edition edition.

Genschel, U. and Meeker, W. Q. (2010). A comparison of maximum likelihood and median-rank regression for weibull estimation. *Quality Engineering*, 22(4):236–255.

Gutschi, C., Furian, N., Suschnigg, J., Neubacher, D., and Voessner, S. (2019). Log-based predictive maintenance in discrete parts manufacturing. *Procedia CIRP*, 79:528 – 533.

Hoedemakers, J. A. J. (2020). Creating a preventive maintenance concept for new systems using

multi-objective optimization. Master's thesis, Eindhoven University of Technology.

Hussain, S. and Sastry, V. U. K. (1997). Application of genetic algorithm for bin packing. *International Journal of Computer Mathematics*, 63(3-4):203–214.

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

Kallen, M. and Noortwijk, J. (2005). A study towards the application of markovian deterioration processes for bridge maintenance modelling in the netherlands. *Advances in Safety and Reliability - Proceedings of the European Safety and Reliability Conference, ESREL 2005*, 1.

Kans, M. and Ingwald, A. (2008). Common database for cost-effective improvement of maintenance performance. *International Journal of Production Economics*, 113(2):734 – 747. Special Section on Advanced Modeling and Innovative Design of Supply Chain.

Konak, A., Coit, D. W., and Smith, A. E. (2006). Multi-objective optimization using genetic algorithms: A tutorial. *Reliability Engineering & System Safety*, 91(9):992–1007. Special Issue - Genetic Algorithms and Reliability.

Koochaki, J., Bokhorst, J., Wortmann, H., and Klingenberg, W. (2012). Condition based maintenance in the context of opportunistic maintenance. *International Journal of Production Research - INT J PROD RES*, 50:1–12.

Kundu, D. (2008). Bayesian inference and life testing plan for the weibull distribution in presence of progressive censoring. *Technometrics*, 50(2):144–154.

Laggoune, R., Chateauneuf, A., and Aissani, D. (2009). Opportunistic policy for optimal preventive maintenance of a multi-component system in continuous operating units. *Computers & Chemical Engineering*, 33(9):1499–1510.

Liu, Y., Chen, Y., and Jiang, T. (2020). Dynamic selective maintenance optimization for multi-state systems over a finite horizon: A deep reinforcement learning approach. *European Journal of Operational Research*, 283(1):166–181. cited By 19.

Marler, R. T. and Arora, J. S. (2004). Survey of multi-objective optimization methods for engineering. *Structural and multidisciplinary optimization*, 26(6):369–395.

Messac, A. and Mattson, C. (2002). Generating well-distributed sets of pareto points for engineering design using physical programming. *Optimization and Engineering*, 3:431–450.

Murata, T., Ishibuchi, H., et al. (1995). Moga: multi-objective genetic algorithms. In *IEEE international conference on evolutionary computation*, volume 1, pages 289–294.

Nicolai, R. and Dekker, R. (2008). Optimal maintenance of multi-component systems: A review. *Springer Series in Reliability Engineering*, 8:263–286.

Okogbaa, G., Huang, J., and Shell, R. L. (1992). Database design for predictive preventive maintenance system of automated manufacturing system. *Computers & Industrial Engineering*, 23(1):7 – 10.

Pandey, M., Zuo, M. J., and Moghaddass, R. (2016). Selective maintenance scheduling over a finite planning horizon. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, 230(2):162–177.

Peng, H. and Zhu, Q. (2017). Approximate evaluation of average downtime under an integrated approach of opportunistic maintenance for multi-component systems. *Computers & Industrial Engineering*, 109:335 – 346.

Pintelon, L. and Parodi-Herz, A. (2008). Maintenance: An evolutionary perspective. In *Complex system maintenance handbook*, pages 21–48. Springer.

Reeves, C. R. (2010). *Genetic Algorithms*, pages 109–139. Springer US, Boston, MA.

Shafiee, M. (2015). Maintenance strategy selection problem: an mcdm overview. *Journal of Quality in Maintenance Engineering*.

Shi, Y., Zhu, W., Xiang, Y., and Feng, Q. (2020). Condition-based maintenance optimization for multi-component systems subject to a system reliability requirement. *Reliability Engineering & System Safety*, 202:107042.

Soland, R. M. (1969). Bayesian analysis of the weibull process with unknown scale and shape parameters. *IEEE Transactions on Reliability*, R-18(4):181–184.

Tiddens, W. (2018). *Setting sail towards predictive maintenance: developing tools to conquer difficulties in the implementation of maintenance analytics*. PhD thesis, University of Twente, Netherlands.

Timmermans, B. (2012). *Development and application of a decision model for synchronizing condition-based maintenance at ASML*. PhD thesis, Master's thesis Eindhoven University of Technology the Netherlands.

Tinga, T. (2010). Application of physical failure models to enable usage and load based maintenance. *Reliability Engineering and System Safety*, 95(10):1061–1075.

Van Horenbeek, A. and Pintelon, L. (2013). A dynamic predictive maintenance policy for complex multi-component systems. *Reliability engineering and system safety*, 120:39–50.

van Noortwijk, J. M. (2009). A survey of the application of gamma processes in maintenance. *Reliability Engineering & System Safety*, 94(1):2–21.

Vandermerwe, S. and Rada, J. (1988). Servitization of business: Adding value by adding services. *European Management Journal*, 6(4):314–324.

Vu, H. C., Do, P., Barros, A., and Bérenguer, C. (2014). Maintenance grouping strategy for multi-component systems with dynamic contexts. *Reliability Engineering & System Safety*, 132:233–249.

Walter, G. and Flapper, S. D. (2017). Condition-based maintenance for complex systems based on current component status and bayesian updating of component reliability. *Reliability Engineering & System Safety*, 168:227 – 239. Maintenance Modelling.

Wang, J., Li, C., Han, S., Sarkar, S., and Zhou, X. (2017). Predictive maintenance based on event-log analysis: A case study. *IBM Journal of Research and Development*, 61(1):11–121.

Wildeman, R., Dekker, R., and Smit, A. (1997). A dynamic policy for grouping maintenance activities. *European Journal of Operational Research*, 99(3):530–551.

Wu, T., Yang, L., Ma, X., Zhang, Z., and Zhao, Y. (2020). Dynamic maintenance strategy with iteratively updated group information. *Reliability Engineering & System Safety*, 197:106820.

Xu, J., Liang, Z., Li, Y.-F., and Wang, K. (2021). Generalized condition-based maintenance optimization for multi-component systems considering stochastic dependency and imperfect maintenance. *Reliability Engineering & System Safety*, 211:107592.

Zhao, H., Xu, F., Liang, B., Zhang, J., and Song, P. (2019). A condition-based opportunistic maintenance strategy for multi-component system. *Structural Health Monitoring*, 18(1):270–283.

Zhu, Q. (2015). *Maintenance optimization for multi-component systems under condition monitoring*. PhD thesis, Industrial Engineering & Innovation Sciences.

Zitzler, E. and Thiele, L. (1999). Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE transactions on Evolutionary Computation*, 3(4):257–271.

Appendix A

Bayesian updating implementation verification

In this Appendix, the Bayesian updating model created in Python with the PyMC3 package is verified. This verification is performed with two test data sets from Ebeling (2019) and one simulated data set. The different sections in this Appendix show the results of the Monte Carlo Bayesian updating model verification with different data sets. In all verification's a Monte Carlo simulation with 10.000 simulations is performed.

A.1 Verification one

The data used as input in this verification comes from Chapter 15 of Ebeling (2019) (Example 15.10). The failure data from Table A.1 is a result of testing 40 units for 365 days. There are two failures modes of failing, mode A and mode B. Besides that, 23 of the 40 units have not failed in the 365 testing days. These 23 units are censored data points. For the Bayesian updating of the Weibull distribution of failure mode A, all failure points B and the 23 survived units are used as censored data. For updating failure mode B, all failure point A and the 23 survived units are used as censored data.

We assume that we already know the shape parameter and that the scale is Uniform distributed with a value between 0 and 6000 for failure mode A. The scale parameter is Uniform distributed between 0 and 1750 for failure mode B and all failure data.

Table A.1: Data form Example 15.10 of Ebeling (2019)

Failure moments	2	8	15	30	35	61	123	123	132	184	186	202	218	232	269	297	333
Failure mode	A	A	A	A	A	B	A	B	B	A	A	A	A	B	B	B	A

From the Monte Carlo simulation for failure mode A, we found a scale parameter of 3741. A least-squares distribution fit for failure mode A resulted in a scale parameter of 3732.7 in (Ebeling, 2019). The Monte Carlo simulation result is very close to this parameter.

The Monte Carlo simulation for failure mode B resulted in a scale parameter of 1129. The least-squares method resulted in a scale parameter of 1037.6. This means that the Monte Carlo simulation results are relatively close to the least-squares distribution fit.

From the Monte Carlo simulation for both failure modes, we found a scale parameter of 1078. We used only the 23 units that survived 365 days as censored data and both failure modes as failures in this situation. The least-squares distribution fit resulted in a scale parameter of 1083.6. The result obtained with the Monte Carlo simulation is very close to the distribution fit results.

A.2 Verification two

The data used as input in this verification comes from Chapter 15 of Ebeling (2019) (Example 15.17). As prior values of the shape and scale parameters, a Uniform distribution is used. The shape is Uniform distributed with a value between 0.5 and 2.5 and the scale is Uniform distributed with a value between 200 and 600. In this example, both censored as uncensored data is used in the input data. This data is shown in Table A.2.

Table A.2: Data form Example 15.17 of Ebeling (2019)

Failure moments	34	136	154	189	286	287	334	353
Censored moments	145	200	380	500	500	500	500	

The data from Table A.2 is generated from a Weibull distribution with a shape of 1.5 and a scale parameter of 400 with arbitrary censored moments. From the Monte Carlo simulation, we found a shape and scale parameter of 1.51 and 474, respectively. This gives a better result than estimating the parameters with MLE. The MLE method in Ebeling (2019) resulted in a shape parameter of 1.43 and a scale parameter of 491.

A.3 Verification three

The last verification we performed for the Bayesian updating model is random simulated values by a Weibull distribution. These simulated values are obtained from a Weibull distribution with a shape parameter of 3.5 and a scale parameter of 8.8. The prior shape parameter is Uniform distributed with a value between $\beta - 1$ and $\beta + 1$ and the prior scale parameter is Uniform distributed with a value between 0 and 30.

First, 500 random failure points were simulated as input data for the Bayesian updating. This resulted in a shape of 3.36 and a scale of 8.71. These results are very close to the predetermined values of the shape and the scale parameter.

Another data set with 1000 random moments, including 500 arbitrary failure moments, was simulated as input data. The Monte Carlo simulation gave a shape of 3.38 and a scale 10.71 of as a result. These parameters are a little further away from the initial parameters. However, the parameters are still relatively close to the input parameters, which leads to the conclusion that the method is well implemented.

Verification conclusion

All three verifications with different types of data resulted respectively close estimated Weibull parameters. Even with a small data set including censored data, the Monte Carlo simulation of the Bayesian updating model finds parameters of the Weibull that are closer to the true parameters than with the MLE. All verification results show that Monte Carlo simulation of the Bayesian updating model is well implemented and therefore applicable in our study.

Appendix B

Simulation model verification and implementation

This appendix presents the verification of the single-component maintenance models for the CBM and ABM maintenance policy. The verification of the single-component simulation models is performed in Python. The simulation results are compared with the numerical examples from Zhu (2015) and Hoedemakers (2020). Section B.1 shows the verification results of a CBM model. The verification of the ABM model is presented in section B.2.

B.1 CBM verification

In this section, verification of the simulation model of a CBM component is presented. This section is divided into two verifications. Section B.1.1 shows the verification results with the input parameters from Zhu (2015). The verification results with the input parameter from Hoedemakers (2020) are given in Section B.1.2.

B.1.1 CBM verification input from Zhu

The Gamma process parameters from the numerical example in chapter 3 of Zhu (2015) are used in this verification. For our simulation model, we use the same input parameters as used by Zhu. These input parameters are presented in Table B.1. The simulation model of this thesis does not include USD options for PM. Therefore, these are not included in the simulation model. Since the model of Zhu (2015) considers an infinite horizon and our simulation model has a finite horizon, we set the time horizon to $T = 100$ and perform 1000 simulation runs to obtain approximately infinite results in our simulation.

Table B.1: Input parameter CBM simulation model verification (Zhu (2015))

Parameter	Value
c^{PM-SD}	28.8
c^{CM}	44.6
τ	91
γ	0.211
η	1.85
H	88

Because of the differences between the model of Zhu (2015) and our simulation model, we did not reach the same results. By removing the USD option for PM, we have fewer PM activities in our model. This should result in higher costs than Zhu (2015) found since the PM costs are lower

than CM. The control limit is expected to be larger than found by Zhu (2015). This is because a lower control limit is more risk-averse since our simulation model does not contain USD PM actions. A lower control limit will prevent more CM actions since only SD PM happen in our model.

Zhu found the output of €45.09 as cost rate per day optimal control limit / failure percentage (C^*/H) of 85.71%. We found €83.41 as cost rate per day and a C^*/H percentage of 82.95%. A lower percentage means a lower value of control limit C^*, which results in earlier preventive replacements since the control limit is further away from the failure threshold H. These results are in line with the expectations of our results compared to Zhu because of the model adaptions.

B.1.2 CBM verification input from Hoedemakers

In this section, the verification results of Component B are given. Hoedemakers used a RCM to model the degradation of the components. We implemented the RCM in our simulation to verify the results. The input parameters Hoedemakers (2020) used in his research are given in Table B.2. The ι and κ present the Weibull shape and scale parameter of the RCM, respectively. The ϕ presents the acceleration factor used in the RCM.

Table B.2: Input parameter CBM simulation model verification (Hoedemakers (2020))

Parameter	Value
c^{PM}	140.48
c^{CM}	217.10
d^{PM}	0.16
d^{CM}	1.02
τ	0.50
ι	2.31
κ	31.35
ϕ	1.50
H	30.0

The results of Hoedemakers (2020) model and our simulation are presented in Table B.3. We used a time horizon of T = 100 and performed 1000 simulation runs to obtain approximately infinite results in our simulation. From Table B.3, we see that our simulation model gives approximately the same result. When optimizing Component B on downtime, the results are very close to each other. The results when optimizing Component B on costs are a little further apart but still nearly the same. As a result, we can conclude that the implementation was satisfactory and that we can apply it to our case study.

Table B.3: Verification condition-based component model

Parameters		**Optimized on downtime**		**Optimized on cost**	
		Hoedemakers (2020)	Our output	Hoedemakers (2020)	Our output
C_i	(mm)	20.0	20.0	13.0	13.0
$Z_i(C_i)$	(€/yr)	170.98	171.09	155.07	153.59
$V_i(C_i)$	(h/yr)	0.20	0.20	0.43	0.40

B.2 ABM verification

In this section, verification of the simulation model of a ABM component is presented. This section is divided into two verifications. Section B.2.1 shows the verification results with the input

parameters from Zhu (2015). The verification results with the input parameter from Hoedemakers (2020) are given in Section B.2.2.

B.2.1 ABM verification input from Zhu

In this section, the simulation model of a ABM component is presented. We use the same input parameters as used by Zhu (2015) in chapter 4. These input parameters are presented in Table B.4. The simulation model of this thesis does not include USD options for PM. Therefore, these are not included in the simulation model. Since the model of Zhu (2015) considers an infinite horizon and our simulation model has a finite horizon, we set the time horizon to $T = 100$ and perform 1000 simulation runs to obtain approximately infinite results in our simulation.

Table B.4: Input parameter ABM simulation model verification (Zhu (2015))

Parameter	Value
c^{PM-SD}	1
c^{CM}	10
τ	0.2
α	1.129
β	2.101

Since there is a little difference between the model of Zhu (2015) and our simulation model, we will not reach the same results. By removing the USD option for PM, we have less PM activities in our model. This should result in higher costs than Zhu (2015) found since the PM costs are lower than CM. Additionally, this should result in a lower age limit than found by Zhu (2015) since the costs of PM are lower than CM. To prevent the same number of CMs with fewer PM actions, a lower age limit is used.

Zhu found an output of €5.189 as cost rate per day optimal age limit of 0.400 years as a result. We found €5.459 as a cost rate per day and an age limit of 0.100 years. These results are in line with the expectations of our results compared to Zhu because the model adaption to exclude USD PM actions.

B.2.2 ABM verification input from Hoedemakers

In this section, the verification results of Component I are given. Hoedemakers modelled the failure of ABM by a Weibull distribution. The input parameters Hoedemakers (2020) used in his research are given in Table B.5. The β and α present the Weibull shape and scale parameter of Component I, respectively.

Table B.5: Input parameter ABM simulation model verification (Hoedemakers (2020))

Parameter	Value
c^{PM}	22.69
c^{CM}	99.30
d^{PM}	0.03
d^{CM}	0.89
τ	0.50
β	2.50
α	2.32

The results of Hoedemakers (2020) model and our simulation are presented in Table B.6. We used a time horizon of T = 100 and performed 1000 simulation runs to obtain approximately infinite results in our simulation. From Table B.6, we see that our simulation model gives approximately the same result. When optimizing Component I on downtime, the results are very close to each other. The results when optimizing Component I on costs are a little further apart but still nearly the same. Because of these small differences, we can conclude that we successfully implemented the model for our case study.

Table B.6: Verification age-based component model

Parameters		**Optimized on downtime**		**Optimized on cost**	
		Hoedemakers (2020)	Our output	Hoedemakers (2020)	Our output
A_i	(yrs)	0.4	0.4	1.1	1.2
$Z_i(A_i)$	(€/yr)	48.95	48.71	32.55	31.92
$V_i(A_i)$	(h/yr)	0.09	0.09	0.20	0.19

Appendix C

Advised system solutions

In this appendix, advice for the system solutions is given. This advice uses the Pareto frontier of the system solutions found by the GA as starting point. The Pareto frontier of the system solutions is shown in Figure C.1. Decision rules are used to select advised solutions from the Pareto frontier. The following two decision rules are used to filter system solutions from the Pareto frontier:

- Select the solutions with minimal downtime out of all solutions with downtime larger than a defined boundary. This boundary increases with steps of one hour downtime.
- Select the solutions with minimal costs out of all solutions with costs higher than a defined boundary. This boundary increases with steps of one hundred euros.

The first decision rule starts by finding the system solution that gives minimal downtime with a boundary of up to and including 10 hours of downtime. The next selected solution should have more than 11 hours of downtime. This continues until no solutions can be found at a particular boundary. With this decision rule based on downtime, we select three solutions represented with red dots in Figure C.2. The same procedure is carried out to find solutions based on the cost decision rule, resulting in seven solutions represented with blue dots in Figure C.2.

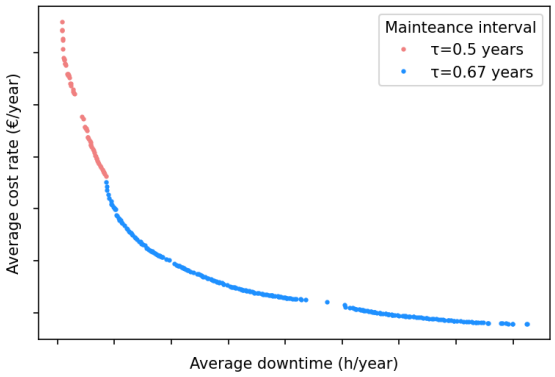

Figure C.1: Pareto optimal system solutions found by the GA

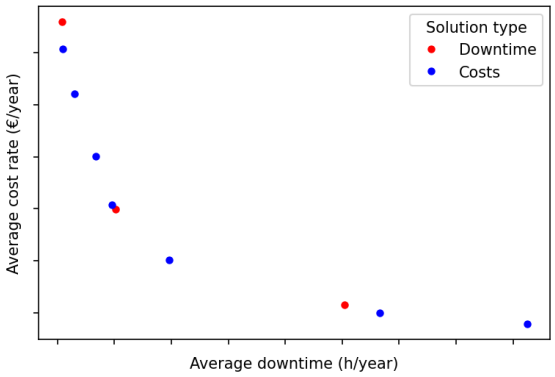

Figure C.2: Filtered system solutions based on downtime and cost decision rules

When we look at Figure C.2, we see all filtered system solutions from the two decision rules by red and blue dots. When we move from the most left red dot to the middle red dot in the figure, we see a steep decrease in system costs and a small increase in system downtime. Since there is just a small increase in downtime and a large decrease in costs, it is advised only to consider the middle red dot when there are no hard restrictions on the maximum allowed downtime of the system. When we move from the most right red dot to the most right blue dot in Figure C.2, we

see a small decrease in costs and a large increase in downtime. It is recommended to only consider the most right red point since a small cost increase results in a large decrease in downtime. This is only recommended when there are no restrictions on the maximum costs of the system.

There is just one blue solution between the two recommended red solutions. These three solutions are recommended to consider for Lely as possible maintenance options. Figure C.3 shows these three options and places them in perspective with the current maintenance schedule. The colour of the three advised system solutions represents the maintenance interval of 0.67 years (8 months) which is used throughout this report.

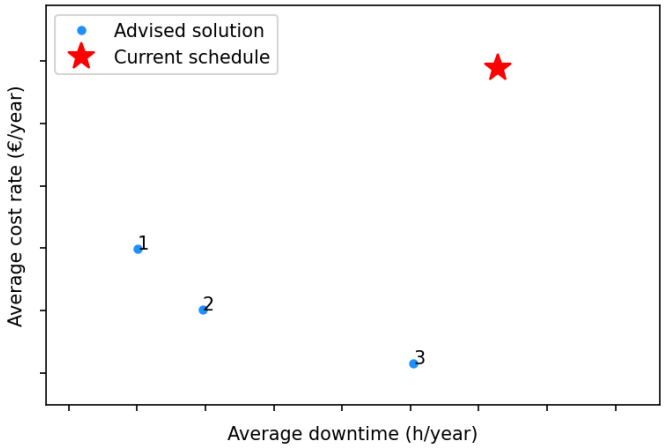

Figure C.3: Advised system solutions compared with the current maintenance schedule

As can be seen from Figure C.3, there is enough space to improve the current maintenance schedule in both maintenance costs or system downtime. The corresponding component maintenance thresholds, downtime and costs of ABM and CBM components are presented in Table C.2 and Table C.1, respectively.

Table C.1: ABM component solutions for advised system solutions

	Solution one			**Solution two**			**Solution three**		
Component name	A_i	$V_i(A_i)$ (h/yr)	$Z_i(A_i)$ (€/yr)	A_i	$V_i(A_i)$ (h/yr)	$Z_i(A_i)$ (€/yr)	A_i	$V_i(A_i)$ (h/yr)	$Z_i(A_i)$ (€/yr)
Component E (2)	0.7	0.14	42.54	0.7	0.14	42.54	0.1	0.22	37.43
Component F (2)	4.9	0.04	14.93	3.9	0.05	12.27	3.9	0.05	12.27
Component G (2)	5.3	0.06	14.28	3.9	0.08	10.75	3.9	0.08	10.75
Component H	2.6	0.08	12.54	2.6	0.08	12.54	3.3	0.08	10.91
Component I	1.0	0.04	18.12	1.0	0.04	18.12	1.4	0.06	14.76
Component J	0.8	0.05	6.99	0.8	0.05	7.00	0.8	0.05	7.00
Component K	1.1	0.06	7.43	1.1	0.06	7.43	1.1	0.06	7.43
Component L (2)	5.2	0.09	18.09	5.2	0.09	18.09	5.7	0.09	17.56
Component M	2.1	0.07	21.30	2.6	0.07	21.02	2.1	0.07	21.29

Table C.2: CBM component solutions for advised system solutions

	Solution one			**Solution two**			**Solution three**		
Component name	C_i	$V_i(C_i)$ (h/yr)	$Z_i(C_i)$ (€/yr)	C_i	$V_i(C_i)$ (h/yr)	$Z_i(C_i)$ (€/yr)	C_i	$V_i(C_i)$ (h/yr)	$Z_i(C_i)$ (€/yr)
Component A	91	0.47	117.15	87	0.48	110.29	83	0.50	106.32
Component B	17	0.49	173.86	14	0.51	164.27	10	0.55	156.84
Component C	62	0.63	75.71	62	0.63	75.71	62	0.63	75.71
Component D	61	0.48	96.29	60	0.48	93.89	59	0.50	92.36