Faculdade de Engenharia da Universidade do Porto

Continuous Maintenance System for optimal scheduling based on real-time machine monitoring

Francisco José Oliveira Costa

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Supervisor: Gil Manuel Gonçalves

Second Supervisor: João Pedro Correia dos Reis

February 24, 2018

Resumo

Hoje em dia, é cada vez mais importante para as empresas optimizar atividades de manutenção devido ao aumento dos custos relacionados com as mesmas actividades.

A necessidade de efectuar manutenções é principalmente causada pelo desgaste dos componentes, que levam à falha total da máquina. Por isso, uma monitorização rigorosa dos processos de produção necessita de ser feita. Com esta base, e para aumentar a eficiência da linha de produção, é necessário monitorizar continuamente o desempenho das máquinas e, juntamente com todos os dados históricos de manutenção, criar estratégias para minimizar os custos relacionados com manutenção.

Portanto, o objetivo principal desta dissertação é estudar uma abordagem para reduzir o custo de manutenção. De modo a resolver este problema, é proposto um modelo conjunto de manutenção preditiva e optimização da taxa de produção de cada máquina.

O bloco de manutenção preditiva é responsável por prever falhas e programar actividades de manutenção, no entanto, quando uma maquina está em manutenção, a mesma não produz, o que poderá comprometer a satisfazer os objectivos de produção. Portanto, o papel do modelo de otimização de carga das máquinas é variar as configurações de carga dos equipamentos para encontrar o equilíbrio ideal entre taxa de produção e taxa de degradação.

Um *Gradient Boosting Classifier* é usado para prever falhas na máquina. As atividades de manutenção são automaticamente agendadas para serem executadas imediatamente antes da ocorrência de uma falha prevista. Para o modelo de otimização, um Algoritmo Genético é implementado.

De modo a avaliar o desempenho da solução proposta, dois testes diferente são feitos. Um utilizando o algoritmo de otimização da taxa de produção e um teste de referência que utilizará apenas o modelo de manutenção preditiva. Para comparar ambas as simulações, métricas, tais como, OEE e disponibilidade serão analisadas.

Keywords: Manutenção Predictiva, *Gradient Boosting Classifier*, Algoritmo Genético, Taxa de Produção

Abstract

Nowadays, the maintenance activities are the ones that most draw the attention of companies due to the increased costs of sudden machines stop, and consequently, stop the production processes.

These stops are mostly caused by wear-out of its components that lead to machine breakdown and a close monitoring of the manufacturing processes need to be made. Based on this, and to increase the production line efficiency, there's a need to continuously monitor the machines' performance, and together with all the historical maintenance data, create strategies to minimize the maintenance phases and costs.

Therefore, the main goal of this dissertation is to study an approach to reduce maintenance cost. A joint predictive maintenance and throughput optimization model is proposed.

Predictive maintenance block is responsible to predict failures and schedule maintenance activities, though, while assets are being maintained the total system production rate decreases. Which could possible compromise to satisfy production objectives. Therefore, the role of the throughput optimization model is to, vary assets load settings in order to find the optimum balance between production rate and degradation rate.

A Gradient Boosting Classifier is used to predict machine failures. Maintenance activities are automatically schedule to be performed right before a predicted failure occur. For the optimization model a Genetic Algorithm is implemented.

To evaluate the performance of the proposed solution, two different set-up's will be tested. One making use of the throughput optimization algorithm and one reference test only using the predictive maintenance classifier. To compare both simulations, metrics, such as, OEE and availability will be analysed.

Keywords: Predictive Maintenance, Gradient Boosting Classifier, Genetic Algorithm, Throughout rate

Acknowledgements

This work wouldn't have been concluded without the help of the people that supported me through this project. My sincere gratitude to my supervisor, Gil Manuel Gonçalves, for the challenge that was this project and the help provided, which was key for this dissertation to reach its conclusion. Also want to express my sincere gratitude to my second supervisor, João Reis, for all the help and feedback throughout the whole project and constant presence, and patience whilst allowing me to find my way.

To all my close friends who have been with me throughout this journey and deeply contributed to who I am today. A special thanks to: Joana Lopes, João Correia, Nuno Campos and Rubens Figueiredo.

Finally, but not less important, my sincere gratitude to my family, that did everything tin their reach so I could achieve this goal and constantly support me through everything.

Francisco Costa

"If you steal from one author,
it's plagiarism, but,
if you steal from many,
it's research."

Wilson Mizner

Contents

List of Figures

List of Tables

Abbreviations

AI	Artificial Intelligence
ANN	Artificial Neural Network
ARIMA	Auto-Regressive Integrated Moving Average
ARMA	Auto-Regressive Moving Average
BN	Bayesian Network
CBM	Condition Based Maintenance
CM	Corrective Maintenance
CNC	Computer-Numerically-Controlled
DAG	Directed Acyclic Graph
DBN	Dynamic Bayesian Network
DEAP	Distributed Evolutionary Algorithms in Python
EKF	Extended Kalman Filter
FMS	Flexible Manufacturing System
GA	Genetic Algorithm
HMM	Hidden Markov Models
HSMM	Hidden Semi-Markov Model
KF	Kalman Filter
IMS	Intelligent Manufacturing Systems
MILP	Mixed Integer Linear Programming
ML	Machine Learning
MPP	Master Production Plan
MTBF	Mean Time Between Failures
MTTF	Mean Time To Failure
MTTR	Mean Time To Repair
OEE	Overall Equipment Effectiveness
OP	Opportunistic Maintenance
PDF	Probability Density Function
PM	Preventive Maintenance
RMS	Reconfigurable Manufacturing Systems
RUL	Remaining Useful Life

Chapter 1

Introduction

The following chapter intends to clarify and define the scope of this dissertation. Providing a context to the subject in study, as well as, the motivation behind the proposed topic. Thus, this chapter aims to present the problem in study, the main goals to be reached, as well as the structure of the document.

1.1 Context and Motivation

Due to the economic turmoil and to the increasing competition in the markets, companies are seeking new ways to maximize their profits. To remain competitive in such harsh environments, industries need to reduce their production costs, where maintenance is one of the most critical operations, representing about 15 to 70% of the expenses [1]. Bearing that in mind, maintenance is often regarded as a cost driving necessity rather than a competitive resource, being a profit-generator [2].

Thus, with the motivation to improve maintenance strategies, it is noticeable a shift from more simple strategies, such as corrective maintenance and preventive maintenance, to more advanced condition based maintenance[3].

As new manufacturing polices are being employed such as lean concepts and just-in-time process, unexpected breakdowns can drive to great losses that derive from different causes: lost production, failed shipping schedules, and poor customer satisfaction. So, with that in mind it's extremely necessary to continuously inspect the current state of the assets, and to accurately predict the Remaining Useful Life (RUL) of the equipment[4].

Thus, predicting RUL and to properly schedule maintenance activities of a production line are key factors for a company to be profitable and competitive. Hence, a broad amount of approaches have been studied in literature. However, after researching, was found that only few authors explored the benefits of combining predictive maintenance with machine throughput optimization to minimize maintenance effects on production targets[5, 6]. Therefore, this dissertation proposes to explore the benefits of the mentioned approach.

1.2 Goals

The main goal of this dissertation is to develop a framework which reduces the maintenance costs of a production line, integrating a prognosis technique to predict the time to failure of each machine, and adapt the maintenance scheduling approach in order to optimize the number of maintenance activities to perform together with the production throughput rate. In discussion is a parallel production line characterized by having:

- Weekly production targets
- Negligible set-up time between shifts
- Throughput rate optimization

To access the condition of an asset, data from various sensors must be obtained through continuous monitoring. Then, the data is processed, providing an estimated time to failure and signalising if a maintenance activity shall be scheduled. If a failure is expected to happen in the near future, a maintenance action must be appointed to the time-period prior to the prediction. Each maintenance activity is set to last a defined period of time, depending on which machine is being repaired. And logically, a machine which is undergoing maintenance, does not produce during that shift. Hence, to comply with the master production plan, which sets the weekly production targets, increasing the load of other the machines might be necessary. Hence, increasing the load of a machine implies increasing its hazard rate, which ultimately leads to an accelerated equipment degradation.

To undertake this issue, a proper symbiosis between prognosis and scheduling is pivotal. After a predicted failure and when the scheduling plan appoints one maintenance activity, it updates the system production rate and if needed increases the load of equipment. With this information a new failure prediction is obtained, and it cycles again to the scheduling block. This cycle runs until the Master Production Plan (MPP) is satisfied and the number of maintenance actions is minimized. Therefore, the main target is to optimize the balance between load and maintenance cost, always considering to meet the MPP requirements.

1.3 Structure

After the Introduction, this dissertation contains 5 more chapters. In chapter 2, the literature review is described, as well as previous work, with special focus on Reliability concepts, Degradation models approaches and Maintenance optimization models. In chapter 3, is presented the problem to be solved, the solution conception and how the validation experiment will be evaluated. In chapter 4, the implementation process is described. In chapter 5, the validation experiment is described and its results analysed. In chapter 6 there is the conclusion to this document and the future work is presented.

Chapter 2

Literature Review

In the following chapter, the literature review of this dissertation key topics is presented. First, a general overview of manufacturing systems is provided. Then, moving towards the goal of the dissertation, to understand "How" and "Why" machine failures occur, Reliability concepts are characterized.

In order to improve reliability of production systems, one key step is to analyse the degradation path of failures, and predict them. Therefore, several Life Distributions and Degradation Models are reviewed.

Lastly, maintenance concepts are explained and maintenance scheduling optimization models are reviewed.

2.1 Manufacturing Systems

During the last decades, due to economic globalization and excessive production, manufacturing enterprises are forced to reduce production cost without decreasing quality and flexibility [7].Therefore, companies are forced to reconsider their production paradigms, so that a manufacturing system can be designed and operated efficiently in a ever-changing environment [8] (Figure 2.1[9]).

The first paradigm was Craft Production, which created the product that a costumer requested, but, at a higher cost. Afterwards, the emergence of Henry Ford's invention of the moving assembly line in 1913, marked the beginning of the Mass Production paradigm [10]. Due to economy of scale, it was now possible to produce a huge amount of products with lower cost. However, customization of products was very limited.

As costumers requested for more customizable products, the manufacturing paradigm shifted towards Mass Customization. Where manufacturer's design the basic product architecture and a set of optional features, then, based on the basic architecture costumers are allowed to select the optional features to come up with a personlized product based on the custmer needs [10].

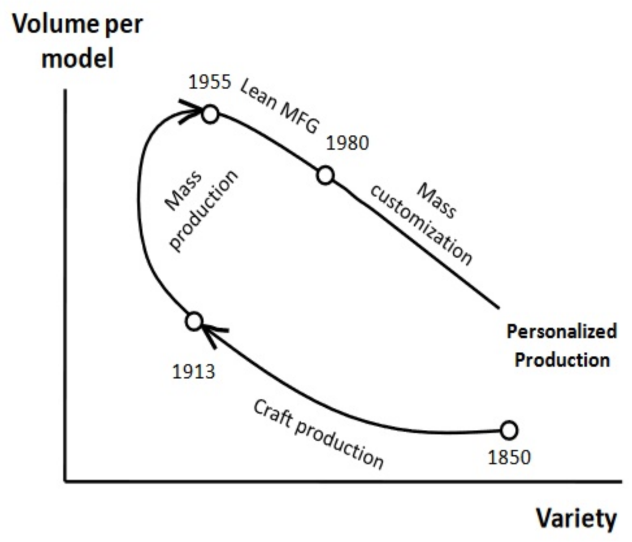


Figure 2.1: Volume variety relationship in manufacturing paradigms

Nowadays, mass customization is of utmost importance since, now more than ever, costumers request individualized customization [11]. This puts an enormous pressure under manufactures, who would have to comply with all environment requirements in order to survive the fierce market competition. Currently, these requirements are understood to be as following [12]:

- Short lead time
- More variants
- Low and fluctuating volumes
- Low price

As paradigms move through Mass-Production to Mass-Customization, so do manufacturing system's configurations. Which, at the beginning consisted of a basic assembly line [13], to more complex paradigms such as Reconfigurable Manufacturing Systems (RMS) and Intelligent Manufacturing Systems (IMS). The following section provides a brief description of different paradigms currently studied in literature.

Flexible Manufacturing System

Flexible Manufacturing Systems (FMS) were first introduced in the 1980's[13]. Typically FMS consist in the integration of general-purpose computer-numerically-controlled (CNC) machines and other programmable forms of automation in assembly lines [14]. Which, allow the manufacturer to program them, in order to produce diferent components, with shortened changeover, in the same system [12].

However, the major drawbacks of FMS are the inability to respond to abrupt market fluctuations and a low throughput rate [15]

Reconfigurable Manufacturing System

To overcome FMS issues, reconfigurable Manufacturing Systems (RMS) was introduced in the 1990's [14]. RMS's are designed to allow rapid changes in the manufacturing structure, as well as in hardware and software components, in order to quickly adjust production capacity and functionality within a part family in response to sudden changes in market or in regulatory requirements.[8]

Intelligent Manufacturing Systems

Intelligent Manufacturing System's (IMS's) are those performing the manufacturing functions as if the human operators are doing the job [16]. They are designed to introduce more agility to manufacturing systems. By collecting data they are able to make "on the fly" changes to manufacturing processes[17]. Such manufacturing systems need to satisfy the following requirements[18]:

- Full integration of heterogeneous software and hardware systems within an enterprise, a virtual enterprise, or across a supply chain.
- Open system architecture to accommodate new subsystems (software or hardware) or dismantle existing subsystems "on the fly".
- Efficient and effective communication and cooperation among departments within an enterprise and among enterprises.
- Embodiment of human factors into manufacturing systems.
- Quick response to external order changes and unexpected disturbances from both internal and external manufacturing environments.
- Fault tolerance both at the system level and at the subsystem level so as to detect and recover from system failures and minimize their impacts on the working environment.

One key requirement of IMS is to minimize the impact of system failures. This feature is of high importance since maintenance related costs represent a high volume of manufacturing systems total costs [1]. Therefore, reliability assessment is a highly important procedure to reduce production costs in a manufacturing system

2.2 Reliability

It is indisputable the influence of Reliability in everyday life. From the common man, who buys products from a retail shop with the expectation that his goods will work properly for a reasonable period of time, to the product manufacturer, who has to have highly reliable machines in order to maintain the same quality standards while minimizing the costs of production.

Since the early ages of humanity, humans started to make tools to aid our physical efforts. Since then, we have been seeking for more reliable and durable equipments. Therefore, as times passed by, new manufacturing techniques (e.g. production lines), were developed. Though, the perfect hotbed for industry development came up with the end of World War 2. As the establishment of mass-production and statistics methods started to transpose to the civil industry, reliability engineering was due to emerge. The catalyst was the introduction of highly complex electronic systems, particularly the vacuum tube, which at the time were considered highly unreliable[19]. Hence, with the unreliability of the devices and the rising need to make electronic equipments, manufacturers started to invest more capital in production quality monitoring. And so, reliability engineering came to be a key factor of engineering systems.

The definition of reliability is not unanimous. Yet, the one that gathers more consent throughout literature, states reliability as the probability that a system will perform its required function under the stated condition for a stated period of time[20].

The statement above defines reliability in a rather generic but meaningful way. It depicts the time which systems are designed to last, where, a product can only be deemed as reliable if it has an adequate life time for its propose. Moreover, it sets that a product is regarded as reliable, not only if it is able to properly function until a desired life span. That it must conform with specifications, by which the system is designed. As an example, a bridge needs to handle a certain weight and a generator needs to provide a certain power to the grid, if these specifications are under-dimensioned, the bridge may collapse or a blackout may occur in a city. In contrast if a system is over-dimensioned, the system may be overpriced for its purpose, and there's no need to have a generator that is able to provide power to a couple houses when its propose is to only provide power to a single motor-home.

Not only a system has to be designed to specifications, but also it has to take into account for the condition in which it will be used. A more rough environment, with higher temperatures, higher humidity requires a different configuration than a more neat environment, which takes in account all environment factors.

The aggregation of all these factors makes reliability assessment a crucial factor in industry. Thus, having equipments with low reliability can lead to great economic losses, and in some extreme situation could cost human lives (e.g. Space Shuttle Challenger Disaster[20]).

For this reason, manufacturers seek to minimize the occurrence of problems. And so, in order to improve assets reliability, there is the need to understand "Why" and How" products fail.[21] Bearing in mind the need to understand how an asset degrades over time, engineers came up with several methods to analyse and quantify system reliability.

2.2.1 Reliability Function

Due to the high number of variables involved, quantifying reliability is an uncertain and laborious task. There are several ways to specify reliability, in [20], Failure Rate, Mean Time Between Failures (MTBF), Mean Time to Failure (MTTF) and Mean Time to Repair (MTTR) are presented as reliability metrics. Though, other metrics such as Overall Equipment Effectiveness (OEE), and Availability will also be explored in this document.

The concept of Availability is still very dominating in analysing the efficiency of an asset. This derives from the division between maintenance and production, where maintenance is responsible for machines unavailability, while the production department is responsible for using equipment [22]. Therefore, as in Equation 2.1 Availability is calculated by the ratio between the actual operating time and the planned operating time [23].

$$Availability = \frac{Actual\,operating\,time}{Planned\,Operating\,time} \tag{2.1}$$

Performance evaluates, as described in Equation 2.2, the ratio between the actual number of units produced and the ideal equipment throughput.

$$Performance = \frac{Total\,Pieces \cdot Ideal\,Cycle\,Time}{Total\,Operating\,Hours} \tag{2.2}$$

Quality is used to indicate the proportion of defective production to the total production volume[24], and expressed as in Equation 2.3 [23]:

$$Quality = \frac{Good\,Pieces}{Total\,Pieces} \tag{2.3}$$

OEE was proposed by Nakagima (1988)[25] to evaluate how effectively manufacturing systems are utilized. It is presented as the ratio between theoretical maximum good output over the actual good output during the planed production time Equation [23].

$$OEE = Availability \cdot Performance \cdot Quality \tag{2.4}$$

For failing systems, maintenance activities are big factors contributing for total system downtime [20]. Therefore, to account the total downtime due to maintenance, MTTR, could be calculated. It is expressed as in Equation 2.5 and represents the average time required to repair a failed component [23].

$$MTTR = \frac{\sum(breakdown\,times\,per\,unit\,of\,time)}{Number\,of\,failures\,per\,unit\,of\,time} \tag{2.5}$$

For repairable systems, MTBF is also calculated which is broadly used in literature, as a metric to understand system behaviours. It is defined as the time a system is fully operating excluding all down times. MTBF is calculated by the inverse of the failure rate

(Equation 2.6)[20].Failure rate will be further discussed in this document.

$$MTBF = \frac{1}{\lambda} \tag{2.6}$$

On the other hand, MTTF is calculated for non-repairable systems which reflects the expected time that a system will fail and expressed by the following formula [26]:

$$MTTF = \int_{0}^{\infty} R(t)dt \tag{2.7}$$

Being R(t) the reliability function. From the definition, reliability is the probability that a system will properly function for a determined period of time t. Therefore, it is possible to represent Reliability as:

$$R(t) = 1 - F(t) \tag{2.8}$$

In which F(t) is the probability that a system will fail before the t period of time. This function is also the cumulative density function. Hence, it is possible to deduce the Probability Density Function(PDF) F(t) which describes the shape of the failure distribution[27].

$$f(t) = \frac{dF(t)}{dt} = -\frac{dR(t)}{dt} \tag{2.9}$$

Therefore it is now possible to define R(t) as:

$$R(t) = \int_{t}^{\infty} f(t)dt \tag{2.10}$$

With R(0)=1 and $lim_{t\to\infty}R(t) = 0$

So, in order to solve R(t), there is the need to find a proper F(t) which accurately represents the degradation path of the system. Some of the most used functions will be further discussed in this document.

2.2.2 Hazard Rate

In the previous subsection, the probability that a product will fail before a determined time was explored. Yet, in a real-life scenario to access the instantaneous rate of failure is essential. Which is also designated as the hazard rate or failure rate. It is given by $\lambda(t)$ and expressed as[27] :

$$\lambda(t) = \frac{f(t)}{R(t)} \tag{2.11}$$

This function has three typical forms, increasing, decreasing and constant. These forms represent assets failure behaviour stages. Bath Tube Curve is a representation of machine falure behaviour based on the three different forms, consisting of a decreasing failure rate stage, followed by a constant and increasing failure rates. This representation is a widely used to model the degradation path of numerous systems[28].

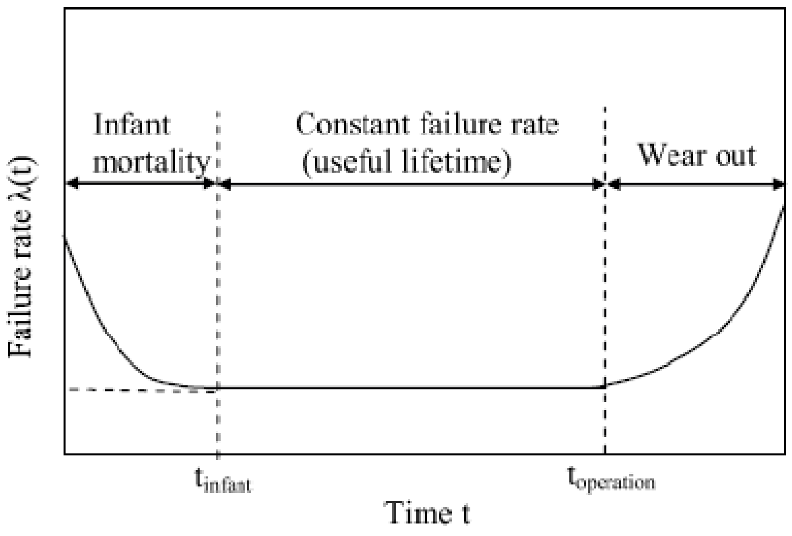


Figure 2.2: Bath Tube Curve

2.2.3 Bath Tube Curve

As previously stated, bath tube curve represents equipment failure behaviour based on three forms of $\lambda(t)$ function. In Figure 2.2[29], the evolution of the hazard rate through time clearly represents the evolution of the different stages that an equipment generally undergoes throughout life-time.

Firstly, it is possible to observe a high hazard rate at the beginning of the product life time which decreases over time.

This stage is called the infant mortality stage. And the high hazard rate is due to various causes such as, inaccurate design to specifications and manufacturing defects [30]. A solution to reduce the impact of this stage is to properly design each product and to perform early failure stress tests at the earliest development phases, to evaluate design weaknesses and uncover specific assembly and materials problems[30] .

After the first stage, the equipment enters the so called useful life period. This stage is characterized by a small and constant hazard rate where the main source of breakdowns are random failures, that may occur due to unpredictable causes.

At the end of the life time, due to ageing the of equipment and cyclical loading among other reasons, the system experiences a wear-out stage. Where it has an increasing hazard rate culminating in the system breakdown. [27]

Delaying this last stage is of high importance to reliability. By doing so, equipment life time is extended. In order to constrict the acceleration of the wear-out, it is compulsory to have a proper maintenance program, to get the most out of each asset.

2.3 Life Distributions

Life Distributions are statistical probability distributions used in reliability engineering to describe the degradation behaviour of different systems. In the following section two of the most used distributions in literature are described, Exponential Distribution and Weibull Distribution.

2.3.1 Exponential Distribution

Exponential Distribution is one of the simplest fault prognostics models. The main characteristics of this distribution are:

- Constant Hazard Rate. Meaning that the components are not subjected to wear overtime[31].
- Memoryless. For that reason, a product that has been operating for many hours has the same probability to failure, than a product which only operated few hours [27].

The PDF of this distribution is given by:

$$f(t) = e^{-\lambda t}\lambda \tag{2.12}$$

Therefore the reliability function is:

$$R(t) = \int_t^{\infty} f(t)dt = e^{-\lambda(t)} \tag{2.13}$$

And as expected the hazard rate is a constant.

$$\lambda(t) = \frac{f(t)}{R(t)} = \lambda \tag{2.14}$$

Being MTBF (Mean Time Between Failure) equal to $\frac{1}{\lambda}$.

The effect's of changing λ are depicted in Figure 2.3[32]. As the value of λ decreases, the PDF stretches in the time domain although it remains always convex[32].

The exponential distribution is a suitable model when components have constant hazard rate, and when at any instance of time the probability of this component to fail is the same. Thus, products with shorter burn-in and wear-out periods and long useful life periods of the bath tube curve, are adequately represented with this distribution [21],[27]. It has been proven that some electronic components, such as transistors and capacitors, follow this distribution, however, for mechanical components where cumulative wear exists through a significant period time, this distribution is not suitable[21], [31].

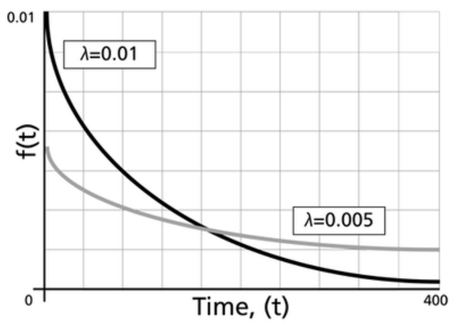


Figure 2.3: Effect of λ on Exponential Distribution PDF

2.3.2 Weibull Distribution

In reliability modeling the most widely used distribution is the Weibull Distribution. This is due to its flexibility, as it can assume various shapes and it can approximate the behaviour of many other distributions[31].

There are three expressions for the Weibull Distribution, 1-parameter,2-parameter and 3-parameter Weibull Distribution, being the latter the most general and with a PDF expression as follows:[33]

$$f(t) = \left(\frac{\beta}{\eta}\right)\left(\frac{t-\gamma}{\eta}\right)^{\beta-1} e^{-\left(\frac{t-\gamma}{\beta}\right)^{\beta}} \tag{2.15}$$

Therefore the reliability function is given by[[33]]

$$R(t) = \int_{t}^{\infty} f(t)dt = e^{-\left(\frac{t-\gamma}{\beta}\right)^{\beta}} \tag{2.16}$$

The hazard is given by:

$$\lambda(t) = \frac{f(t)}{R(t)} = \frac{\beta}{\eta}\left[\frac{t-\gamma}{\eta}\right]^{\beta-1} \tag{2.17}$$

Where η is the scale parameter. As we decrease this parameter, the distribution shrinks in the time scale, otherwise, as we increase this value the distributions is stretched over time(Figure 2.4[26]). γ is the location parameter, refers to the earliest time to failure, if γ>0 it is considered that there are no failures in the system until the value of γ[33] .The shape parameter β,Figure 2.5[26], allows to modify the form of the distribution . With $\beta < 1$, the hazard rate function decreases with time, oppositely, if $\beta > 1$ the hazard rate function increases with time, between $1 < \beta < 2$ there is a faster increase initially in the failure rate and then increases slowly, after$\beta > 2$ increases even more as time passes by.

In literature, Weibull Distribution approaches have been studied for a considerable time span, revealing good results modelling hazard rates [34]. Weibull distribution is applied to estimate the remaining useful life of civil aircraft components. The distribution handled historical failure data in order to accurately estimate the status of components[35].

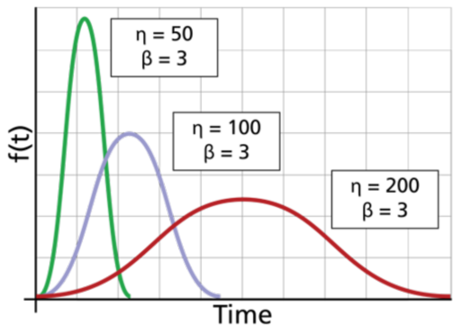


Figure 2.4: Effect of η on Weibull Distribution PDF

It proposed a data-driven prognosis approach to model bearing RUL. A combination of Simplified Fuzzy Adaptive Resonance Theory Map, Artificial Neural Network (ANN) and Weibull distribution are explored. The role of the Weibull distribution is to fit the measurements in the training phase, to avoid areas of fluctu ation in the time domain. The main usage of the distribution is to fit data measurements such as in [36], [37] and [38] which also studied ANN based predictive maintenance approaches, with failure and measurement data fitted by the Weibull distribution. The greatest advantage of using Weibull Distribution, is the flexibility, and the ability to approximate many distributions. The more common approximations using this distribution are: Exponential Distribution (when $\beta = 1$), Rayleigh distribution (when $\beta = 2$) and Normal distribution (when $3 < \beta < 4$) [21].

2.4 Degradation Models

In order to successfully estimate how a system will behave during its life cycle, a proper degradation model needs to be developed. There are various approaches that have been proposed in literature. In this document, the different approaches will be divided into four main categories:

- Experienced-based Models.
- Knowledge-based Models.
- Model-based Approaches.
- Data-driven Models.

2.4.1 Experienced-Based Models

The most simple models are the experience-based, since they are merely based on the probability density function (e.g. Weibull, Exponential, Gamma, Log-normal). Failure and measurement data is acquired and then fitted in a proper distribution, generally,

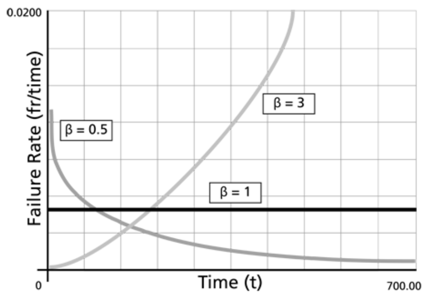


Figure 2.5: Effect of β on Weibull Distribution Hazard Rate

Weibull Distribution [39]. Therefore, this approach is only suitable when a high amount of historical data from identical systems is available. It is a fairly basic model, predicting a failure based on previous failure paths. Hence, it is not able to deal with stochastic failures, nor with dynamic fluctuations of the failure rate (e.g. need to increase the production line throughput, increasing the load of each machine, therefore, increasing the hazard rate) [40].

2.4.2 Knowledge-Based Models

These approaches are developed with the aid of professional human experience. As the system receives new observation data, the model checks for correlation between the observation and the previously developed dataset. With the correlation between the observation and the dataset, the system is able to deduce an output data which can be in form of Remaining Useful Life (RUL)[41]. This approach has two typical examples: Expert systems and Fuzzy Logic systems.

Expert Systems

Expert systems are characterized by having a set of rules that generally come in form of IF condition, THEN consequence[42]. The rules must be developed by an expert, considering that each set of inputs must only have one output value[41].

Since it is a rule based model, the system is not able to properly handle with unexpected scenarios [40]. To deal with unpredictable situations, new rules shall be implemented in the model. Though, as the number of rules increases, so does the computational power needed to model the system [41]. A great disadvantage is that the model is only as good as the experience of the engineer who is in charge of developing the model. Moreover, to scale the problem and increase the number of rules, the model complexity is increased and leads to lack of clarity in the model

Fuzzy Logic

Contrary to Expert systems, where really strict rules are defined, Fuzzy Logic is modelled to deal with imprecise information [42].

Fuzzy Logic consists in a set of overlapping states, membership functions, by which, each input is assigned with truth value (ranging from 0 to 1) correspondent to each membership function. For example, given three states of vibration: no vibration, slight vibration, strong vibration. The asset could be considered to be 0 non vibration, 0.35 slight vibration and 0.7 high vibration. This step is named fuzzification, after it, a set of IF-THEN rules are operated and an aggregate fuzzy set is calculated. The last step is to desfuzzify the aggregate fuzzy set which provides with a single value , that in this case could lead to the overall machine degradation[43].

In literature it is rarely proposed a degradation model operating solely with Fuzzy Logic. Although hybrid methods are highly in vogue, as Artificial Neural Networks are establishing a position in reliability research, more aggregate approaches of ANN and Fuzzy Logic are emerging. Hence, Fuzzy-ANN concepts are the most widely used hybrid solutions [41].

Generally researchers found that hybrid Fuzzy Logic techniques could provide better results than other methods such as Artificial Neural Network. [44] compared Neural-fuzzy approaches and ANN approaches, showing that hybrid methods have better results in life assesment. [45] found as well better results using a neural-fuzzy method to model the deterioration of induction motors.

2.4.3 Model-Based Approaches

Model-Based approaches use a dynamic mathematical representation to monitor a system behaviour [40]. Hence, in order to proper predict an asset RUL an accurate mathematical model of the system is required[46].

Some of the model-based techniques, which will be discussed in this document are:

- ARMA, and subsequent technique ARIMA
- Kalman Filters
- Particle filters

Kalman filters and Particle Filters are based on Bayesian techniques. These methods, rely on residuals calculation to provide with the RUL prediction. Residuals are calculated by taking the difference between the measured variables and their estimate[47]. If the residual is approximately 0, it means that there is no predicted fault. Otherwise with a non-zero residual, a fault is predicted [48].

Kalman Filters

As previously stated Kalman Filters (KF) are a Bayesian technique, which is used to estimate the state of a dynamic system model based on noisy measurements, in order to minimize the mean squared error (Residual) [41].

This approach assumes that a given system is linear, and the measurement noise is Gaussian. Thus, assuming a system as linear could lead to incorrect models, as most real-life systems are non-linear [49].

In regard to this drawback, an alternative non-linear method was developed. Extended Kalman Filter (EKF) is a non-linear version of KF, which uses the linear approximation of the non-linear function, to estimate the current state prediction [50].

Compared to other approaches discussed in this document (ANN, HMM, PF,...), Kalman filter's have a higher computational efficiency which could be useful to model systems with a higher amount of states[41]. Although, if computational efficiency is not the bottleneck of the design, Particle Filters have proven to perform better than Kalman Filters [51]

ARMA Model

The final model-based approach in discussion is the ARMA model, which stands for Autoregressive Moving Average. And is composed by an Auto-Regressive part (AR) and a Moving Average part (MA)[52].

This is a widely used time-series model for RUL prediction [53]. The development of the model consists in three steps which are further explained in [41]:

- Model Identification
- Parameter Estimation
- Model Validation

To be able to assure adequate results, a complete and representative history data is needed, which could be an issue in practical situations, where history data is not always at disposal [52].

Though the model is good at short term prediction of the RUL, long term predictions are not as reliable[41]. To tackle this problem [54] proposes an Auto-Regressive Integrated Moving Average (ARIMA) based model which is able to get better results over long periods.

It is also characterized as being a stationary model, which might be useful when the hazard rate of the system is independent of the age or when the system life-cycle is characterized by an exponential distribution.

2.4.4 Data-Driven Models

Several approaches have been discussed, though, one of the more reliable and noteworthy methods are the data-driven approaches. Therefore, the most used techniques: Artificial Neural Networks (ANN), Bayesian Networks (BN) and Hidden Markov Models (HMM), will be broadly discussed in separate sections.

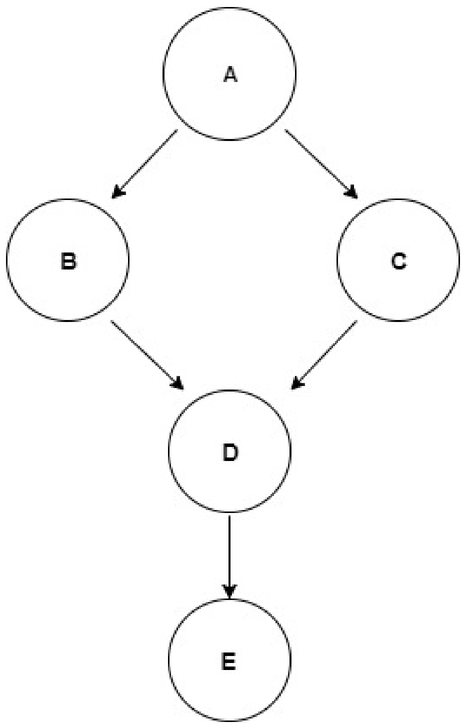


Figure 2.6: Bayesian Network Structure

2.4.5 Bayesian Networks

Bayesian Networks are probabilistic models based on Directed Acyclic Graph's (DAG's)[55]. A directed graph is acyclic if there are no cycles between the nodes [56].

The structure of BN is characterized by having a set of nodes, which represent the system variables, each variable node can be connected by directed arcs to other nodes, which represent the influential relationship between variables [57]

An example of the BN structure is represented in Figure 2.6. Where variable A is considered to be parent of variable B and C, which are both parents of D, and E is the only child of D.

The probability regarding every possible event of a set of random variables X is defined by the joint distribution p(x) [58].Each variable has a joint distribution probability, which is defined by the conditional probabilities given by its parents. For example, the joint distribution probability of node D (Figure 2.6) is given by P(di|ci,bi), where di,ci,bi are the values of the nodes D,C,B respectively. The global joint distribution is given by all conditional distributions probabilities of the system [55].

$$P(X1, X2, ...Xn) = \prod P(xi|pi) \tag{2.18}$$

With pi being the values of Xn parents and xi the values of Xn. As an example, the

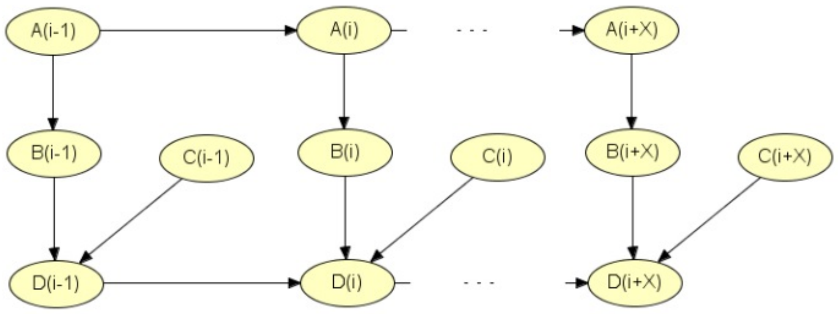


Figure 2.7: Dynamic Bayesian Network Structure

full joint distribution of the Network represented in Figure 2.6 is given by:

$$P(A,B,C,D,E) = p(a)p(b|a)p(c|a)p(d|b)p(d|c)p(e|d) \tag{2.19}$$

There are two different stages when designing a BN, structure learning and parameter estimation. Structure learning consists on finding the DAG which fits the system better. Several techniques are used to this process (e.g log-likelihood), though the most widely used are search algorithms based in heuristics, as the K2 Hill Climbing and Markov Chain Monte Carlo[59].

Next step is, from the observed data, compute the conditional probabilities for each variable and estimate the system parameters. After the inference process, a training process is needed. With that in mind a training data set is used and a wide range of classifiers can be implemented for the training process. Regarding, the classifiers, [60] provides a comprehensive review on the theme. Some of the reviewed classifiers are: Naive Bayesian classifier, TAN networks and Selective Naive Bayesian classifier.

2.4.5.1 Dynamic Bayesian Networks

Dynamic Bayesian Networks(DBN) are similar to Bayesian Networks, although, they have directed BN arcs flow forward in time and are therefore useful for modelling time series data[41].

DBN is defined as "a directed acyclic graphical model of stochastic process. It consist of time-slices (or time steps). With each time-slice contains its own variables"[61].

An example of this model can be represented in Figure 2.7[62], where it is clearly noticeable the different time slices (i-1, i, i+X) and the relationship among the different variables A,B,C,D.

Dynamic Bayesian Network are widely used in literature for prognosis. [63] developed an ISPM model to prognosis, it mainly uses DBN to model a gas turbine compressor system and integrates an Ant Colony algorithm to search for the most reliable fault propagation path. Not all failures may cause a system failure, though many subsequent failures could lead to a machine breakdown.

DBN is used for prognostics to help a decision based maintenance system. Second order time dependencies are applied so, when a "reset" decision is made, the system condition goes back to the previous health state[64]. [65] proposes a framework for maintenance decision making of an airplane. It predicts the wear of a braking system based on BN model, which provides reliable RUL estimations. [66] uses a DBN approach for life extension assessment, to a safety fire water pump ageing system. It uses an on demand, standby approach, where periodic function tests are made to access system functionality. A set of time based maintenance, corrective maintenance and condition based maintenance are used and can only be applied when the function tests are made. Monte Carlo method is used to estimate the conditional probabilities and maximum likelihood method to estimate system parameter is used.

[57] tackles the problem of having continuous inputs with a junction tree approach to discretize variables. [67] accesses the various parameters of the network with a Markov Chain-Monte Carlo method. Lastly [56] predicts the RUL of filter-bags for a carbon black factory. Adopting a delay-time analysis in conjunction with a maintenance modelling technique, to reduce the system downtimes, whilst, optimizing inspection time intervals .

Dynamic Bayesian are a flexible model which can generalize other well known models as Kalman Filters and Particle Filters, which were previously referenced, and Hidden Markov Models and its variants, which will be discussed in the next section.

2.4.5.2 Hidden Markov Model

Hidden Markov Model (HMM) is a statistical approach, to model systems that evolve through a finite number of states based on Markov Chains principle [51].

A Markov Chain is sequence of states, where the probability of the current state p(t) is only dependent of the probability of the previous p(t-1) [68].

$$P_r(X_{t+1} = x|X_0, X_1, ...X_n) = P_r(X_{n+1} = x|X_n) \tag{2.20}$$

It is composed of two stochastic processes. An unobservable Markov Chain, which represents the states of degradation, ergo, they are designated Hidden models. And an observable stochastic process, which is derived by the information collected through continuous monitoring and failure history[69]. [42]and [70] propose a Hidden Markov Model based, prognostics principle which stands as follows:

- Develop and train the HMM for all component health states. Regarding the trained HMM's, it is assumed that the transition time of estimated vectors follow a multivariate distribution.

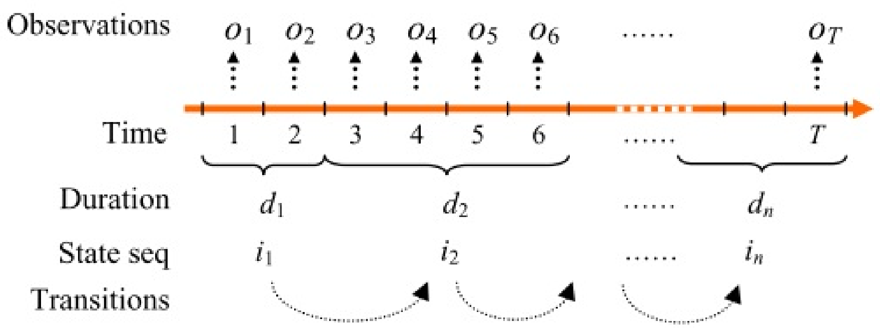


Figure 2.8: HSMM Scheme

- Once the distribution is accessed, the conditional probability distribution of a distinct state transition can be estimated, based on the preceding transition.

The coordinates of the intersection points, of log-likelihood trajectories for different HMM's along the health condition axis represent the estimated "state transition time instants". That is, the probability distribution for the state transition time is estimated based on the estimation of "state transition time instants". The overall shapes of actual log-likelihood plots do not approximate the ideal plots of the estimate based by state transition time instants. Thus, making the estimation of the latter more difficult.

The main advantage of HMM is that as long as enough data is available, it can classify time-series data without knowledge of the problem [41]. They are also easy to interpret, but the fact of being based on Markov Chains assumption, limits the practicability of this technique in real life scenarios, where states are not only influenced by the previous state values [51].

Due to the limitations of Hidden Markov Models, Hidden Semi-Markov Models (HSMM) were developed, since they are not bound to Markov chain assumption they perform better in practical systems.

Unlike HMM, where each state generates a single observation, a state in HSMM is able to generate a segment of observations[68].

A general schema of HSMM is presented in Figure 2.8[71], where each state lasts for a predefined time unit. As for example state 1 lasts two time units and each state, as previously stated, can provide with multiple observations.

The system transits from state to state until the OT, the last observation, is produced.

As well as in HMM, a sufficient amount of data is needed to train the model in order to provide a reliable life forecast.[41]. The amount of data the model consumes to train the HSMM process is another drawback of the approach [72].

Nonetheless, this method is being widely used by researchers to access the residual useful life of a component. [73] proposes an age-dependent model to estimate RUL, based on three types of ageing factors, constant, multiple and exponential ageing factors.[74] also models an ageing-dependent system, though, it integrates Grey model with HSMM

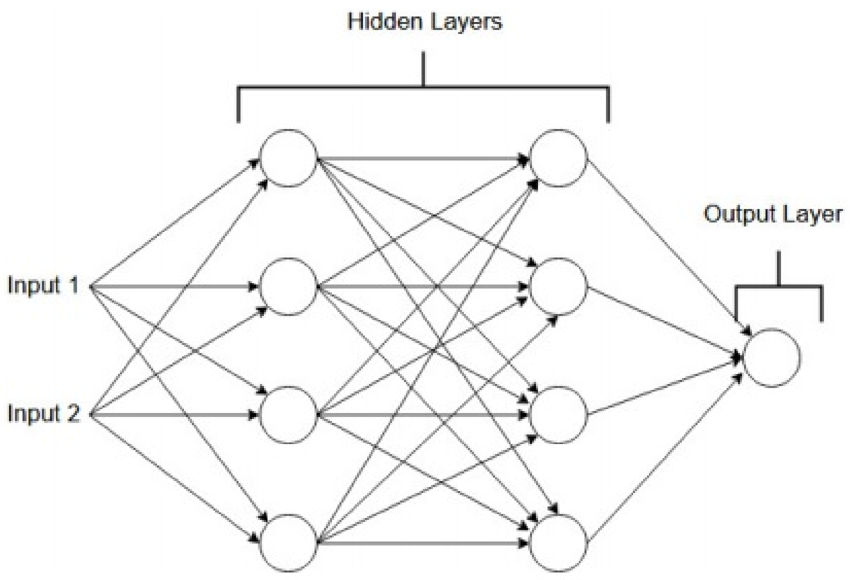


Figure 2.9: ANN Scheme

[68] suggests a jointly HSMM framework for prognosis and diagnosis based on forward-backward inference algorithm. Also, compares HMM and HSMM proving that the latter performed better in the proposed system.

2.4.6 Artificial Neural Network's

Artificial Neural Network's (ANN's) are a data-driven approach which have been gathering more and more recognition to model prognosis problems.

As in other approaches they provide an estimated output result of RUL [75]. Whereas in other models, one of the big pros of using an ANN is the possibility to model complex systems without any knowledge or assumption of the system structure [76]. Thus, it does not require an analytical expression of system behaviour.

Artificial Neural Networks are usually structured in layers: an input layer, one or more hidden layers and an output layer. Depending on the system requirements, the layers consist of one or more nodes (Figure 2.9[71]).

A node is the basic unit of Artificial Neural Networks. Figure 2.10 represent the structure of the node inputs and outputs[77]. The input of each node consists of the sum of the weighted inputs,$\sum W_k.I_k$, and a bias which allows the possibility of shifting the activation function in the horizontal axis.

This sum is the input value of an activation function which processes the data. Sigmoid functions and hyperbolic tangent are the most utilized activation functions [77].

The set of outputs, for each activation function node, are then used as inputs for the subsequent nodes.

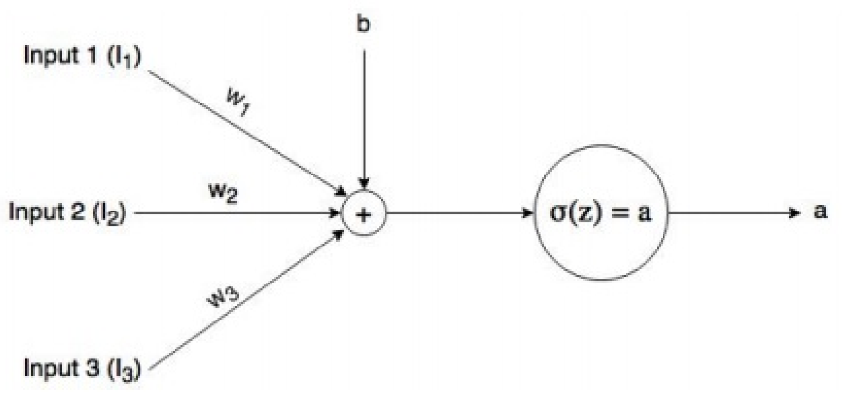


Figure 2.10: NN Node Scheme

Artificial Neural Networks can be classified as feed-forward networks or dynamic networks. In feed-forward networks the output response to a set of inputs is independent from the previous iteration set of outputs. Whereas, in dynamic networks, the output response to a set of inputs might be dependent from the last set of outputs. Hence, they are well suited to model time-variant systems [41].

Some ANN architectures are proposed, as the more conventional for RUL assessment are:

- Multi-layer Perceptron, for feed-forward networks.
- Recurrent-networks, for dynamic networks

As in other models discussed hitherto (e.g. DBN), in order to provide with good estimations, the network must be trained.

Training processes generally require data training sets, to automatically adjust the weights and parameters of the ANN until the output value reach the desired minimum error compared to the validation value.

The main disadvantage of using ANN's to model a system is the lack of transparency on how decisions are reached during training[51]. To improve the efficency of the model, new integrated approaches were explored, mainly neural-fuzzy approaches were used in literature [44, 78]. An ANN designing obstacle is to avoid over-fitting the network during training process. A model is considered overfitted when it does not perform well in real-world cases although it fits the training data well. To avoid this problem an effective validation system is required .

Albeit these limitations, Artificial Neural Networks are the most commercially available prognostics modelling system. Principally, due to the facility of ANN modelling highly and non-linear systems, without physical understanding.[41] Many articles suggest ANN as an approach to efficiently model system Remaining Useful Life. [79] proposes a Condition Based Maintenance (CBM) approach for wind power systems, where an ANN is used to predict the RUL. It assumes the Weibull distribution to model the component useful life time, developing an optimized maintenance schedule based on the failure probabilities provided by the ANN.

[80] come up with a FFNN to access the RUL of bearings in rotating machines. Vibration root mean square and kurtosis values are used as fitted measurements to avoid the noise provided from the measurement data.

[81] explores a Recursive Neural Network approach for prognosis. Proposing a model to access machine health state in the long-term. Proving it efficiency in long-term predictions. [37] modified the approach of [82], to make possible of having unequally spaced inspection points. Inputs are fitted measurements of the condition monitoring data and a genetic algorithm is used to estimate values of the Weibull distribution.

[83] applied ANN to the prediction of mill liners useful life. It used a Multi-Layer Perceptron to model the system, and developed a principle to eliminate input components who had little or no influence to the total variation of the data set. Thus reducing the amount of inputs.

To summarize, Table 2.1, depicts the advantages and disadvantages of using all the aforementioned methods and approaches.

2.5 Maintenance

Each and every factory shop-floor is composed of several machines. As these machines get older, they start a process of deterioration that ultimately leads to the break down of the assets. Consequently, after the machine failure a set of maintenance procedures need to be done, in order to re-establish the normal operation of the production line.

It is undeniable that maintenance plays a big role on manufacturers expenses. As manpower need to be assign to this duty, spare parts and new equipment need to be bought to properly maintain the shop-floor assets. Though, nowadays maintenance starts to be seen as a profit-generator procedure rather than an expense. With improved maintenance plans and procedures being developed, companies are noticing better performance, higher quality rates and enhancing assets availability. Therefore, production costs are reduced, culminating in an increasing in the overall profits[84].

To help with maintenance decisions, several techniques were developed over time. In this section, three techniques will be discussed:

- Corrective Maintenance (CM)
- Preventive Maintenance (PM)
- Condition Based Maintenance (CBM)

Table 2.1: Degradation Models Review

Category	Methods	Advantages	Disadvantages
Experienced-based models	Life Distributions	-Might be used for different operating conditions without the need for recollecting data	- Solely based on historical failure paths. -Static. Not dealing with hazard rate dynamic variations nor stochastic failures.
Knowledge-based models	Expert systems	-Solving problem by mimicking how human expert make decision. -Simple to develop	-Component/system speciality, hard to build model -Significant number of rules required. -Hard to convert domain knowledge to rule. -As good as the expert knowledge
	Fuzzy-Logic	-Can deal with vague, imprecise information. -Model system in continuum mathematics. -Requires fewer rules than expert systems	-Weight decision is a hardwork. -Fuzzy rules, difficult to determine.
Model-Based models	ARMA models	-Accurate and reliable short-term predictions of RUL.	- Long-term predictions of RUL are less reliable. Relaxed with ARIMA -Stationary model.
	Kalman Filters/ Extended Kalman Filters	-High computing efficiency. - Aplicable to real-time Situations	-KF linear system assumption.EKF deal with non-linearity,though, with higher computation cost -Measurement data, required
Data-Driven models	Bayesian Networks	-Well constructed theoretical basis. - Can deal with expert knowldge.	-A lot of historical state transition and failure data are needed.
	HMM/ HSMM	-Reveal the hidden states change processes. - Able to model temporal data. -Provides confidence limits.	-The Markovian assumptions in HMM are not practical in real world. -HSMM relax the assumptions but complicate the model. - High computational cost. -Relies on a failure threshold.
	Artificial Neural Networks	-Model analytically difficult systems. -Accurate and fast online pattern recognition. -Capable to deal with imprecise data. -Fast computation of complex systems	-Requires high amount of data. -Model retraining is needed if operating conditions change. -Blackbox system. -Relies on a failure threshold.

Corrective Maintenance

Corrective Maintenance (CM) is regarded as a maintenance reactive approach [85]. Therefore, a maintenance action is only made when the equipment fails. As the failure times, are unpredictable and unscheduled, each corrective maintenance action is unplanned.

This unpredictable behaviour can induce great maintenance costs for companies such as:

- Unplanned maintenance, lead to unexpected loss of production[85].
- Higher costs to restore the equipment to an appropriate condition under crisis situation[85].
- Insufficient specialized manpower or spare parts at the failure moment, could lead to greater downtimes[86].

To prevent the effects of unexpected failures and reducing the production costs, having an optimal maintenance schedule plan is a must. For that reason, new alternatives to CM were developed. Though, Corrective Maintenance still plays a role in maintenance planning since unexpected failures could happen every moment.

Preventive Maintenance

Unlike CM, Preventive Maintenance (PM) is based on pre-planned maintenance actions. It hangs on, improving equipment condition, thus, an asset is due to maintenance even if it didn't break down. Each PM action is scheduled and uses time trigger or usage trigger to perform signalization of each maintenance[87].

A PM action is time trigger based, when maintenance activities are scheduled in fixed periods of time (e.g. an equipment which is maintained after one week). Otherwise, usage trigger is, for example, the case of cars inspections, which, each car needs to be inspected after a defined number of kilometres.

Preventive Maintenance, promotes maintenance actions in convenient occasions. Which, decreases system downtimes, since workers are expecting the PM action.Moreover, maintenance activities could be scheduled to either, less productive or down-time intervals of the production line. It provides a larger machine life cycle improving equipment efficiency and reducing the overall production costs.

Condition Based Maintenance

Preventive Maintenance is based on an event trigger principle. Although, having fixed maintenance moments it is not always efficient. An equipment could be in perfectly good condition, and nonetheless, be subjected to maintenance. Lastly, Condition Based Maintenance (CBM) is considered to be more efficient than PM and CM. Maintenance actions are made based on the current condition of the equipment. Hence, CBM optimizes the number of maintenance actions.

The heart of condition based maintenance is condition monitoring [88]. Which, by continuously monitoring equipment signals, such as: lubricating oil level, vibration and temperature. Provides real-time assessment of each asset condition. Condition monitoring values are further used to help optimizing maintenance decisions.

Although, evaluating the system current condition is no easy task, some possible approaches were discussed in the previous section of this document. Nevertheless, none of which, will provide with an absolute solution. [89] refers that the main disadvantages of condition monitoring are: possibility of inaccurate information, that may contain noise, providing imprecise condition estimation; need to have special devices to monitor the equipment, which adds to maintenance expenses.

Imperfect Maintenance

When modelling complex and important equipments, where, a certain degree of reliability is required or when high maintenance costs are involved. The advantages of CBM to other maintenance procedures, largely outweigh the disadvantages. Many CBM and PM models deem replacement as the only maintenance action. After each action, the system returns to a state considered to be " as good as new"[90]. Though, it is easily perceived that in a real life scenario,fully replacement is not always needed.

Acknowledging that, many papers [90], [91], [92] propose an imperfect maintenance approach. Consisting in maintenance actions, which, do not fully restore the equipment life. [91] and [93] consider three kinds of possible maintenance actions:

- Minimal PM
- Imperfect PM
- Perfect PM

In this document, simple maintenance services like, lubricating, tightening loose parts and cleaning dust, are considered as minimal PM. Minimal PM actions do not change the reliability or condition state of the system. Instead it helps to maintain the current state, by slowing down the degradation rate of the system [91]. After a minimal PM action the system is considered to remain in a "as bad as old" state [94].

Imperfect PM induces a reliability improvement factor to the equipment [95]. An equipment goes through imperfect maintenance action when, for example, there is a component part replacement. It does not fully restore the equipment condition but improves the current condition of an asset. Thus, it lies between "as bad as old" and "as good as new".

[96] and [90] propose an hybrid maintenance approach. It combines age reduction of imperfect PM and the hazard rate adjustment of minimal PM (Figure 2.11[96]).

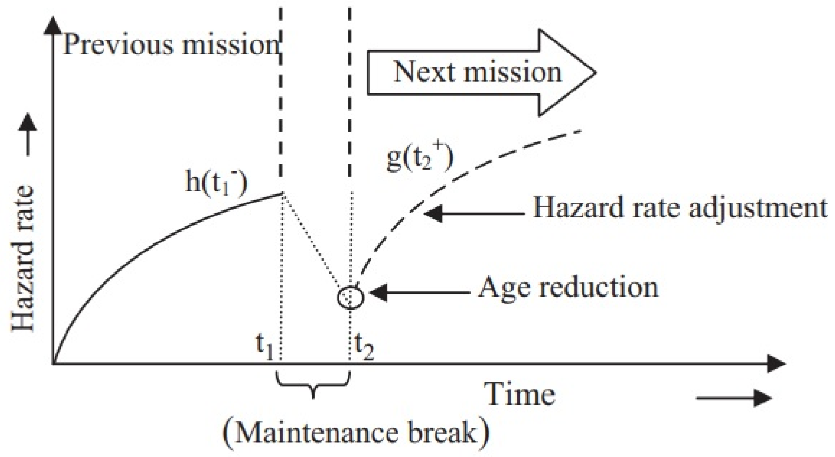


Figure 2.11: Hybrid Imperfect Maintenance

Perfect maintenance is the most costly action, since it fully restores the system to a "as good as new". It is usually employed to key equipment for the whole system, or to components which undergone multiple repairs [92].

To maximize the availability and reducing the costs, maintenance engineers seek for a perfect balance between all PM actions.

Multi-Component Maintenance

Generally in literature it is assumed that after a predefined condition threshold, a maintenance action shall be performed [62]. Although, when scheduling a multi-component production line, sometimes it is appropriate to perform a maintenance action before the predefined threshold.

For example, a production line layout consisting of two machines in series. Supposing that if the first machine is turned off, the following machine has to be stopped as well. In this situation even though the second machine it is not at the maintenance threshold, it might be desirable to perform a maintenance activities, since the line is forced to stop. Thus, reducing system downtimes in the future.

Regarding multi-component maintenance scheduling two approaches are generally recognized in literature [6]:

- Group Maintenance
- Opportunistic Maintenance

Group maintenance as the name suggest it is characterized by grouping a set of assets. When one of the elements of the group is maintained the other elements go through maintenance activities as well [97][86].

On the other hand, an Opportunistic Maintenance (OP) [98] is based on analysing if it is beneficial to maintain any other elements of the shop-floor [99][100]. This benefits could be either economical or functional advantages. The previous example, it is a clear case of a functional opportunistic maintenance. Since the failure/maintenance of one equipment is inherent to the system behaviour. It is opportune that the machine in idle, waiting for another machine to be maintained, also be maintained.

An example of a economical advantageous opportunistic maintenance is to maintain an equipment in a wind farm[101]. Usually, specialized teams need to travel to the wind farm in order to maintain one equipment. Though, with the implicit costs of gathering the team and travelling costs it might be profitable to maintain more than one component for each visit to the wind farm.

2.5.1 Scheduling

In order to optimize maintenance activities and to minimize the costs due to maintenance. An optimal maintenance schedule must be developed. To be able to, optimally schedule maintenance activities, of complex systems, there is the need to aggregate all the previously discussed approaches. Another key element is to build one or more system objective functions, which need to be either maximized or minimized [102].

Usually, the objective functions are either cost driven or reliability driven functions[103]. Logically, every company seeks to maximize it products reliability thus, minimizing the production costs. Reliability centred objective function are more prominent in high risk facilities or safety systems where reliability is a key factor (e.g. Nuclear plant)[89].

To optimize the objective function, several techniques are proposed. [104] listed and reviewed four heuristic techniques:

- Genetic Algorithm (GA)
- Simulated Annealing (SA)
- Tabu Search
- Hybrid approach (Tabu Search and SA)

[105] exposed thirteen different optimization models including GA, an analytical model and simulation models (e.g. Monte Carlo simulation).[103] Divided scheduling techniques in mathematical approaches and heuristics methods. A considerable amount of papers were studied and, among mathematical approaches a Mixed Integer Linear Programming (MILP) approach was the most prominent. Some of this MILP models combined branch-and-bound with simplex method or interior point method, to improve their effectiveness. The heuristic method, which revealed to be the more widely used in literature among maintenance scheduling researches, was the Genetic Algorithm. This document will focus on this method, Genetic Algorithm, which proved to be effective and robust, albeit, rarely providing the optimal solution.

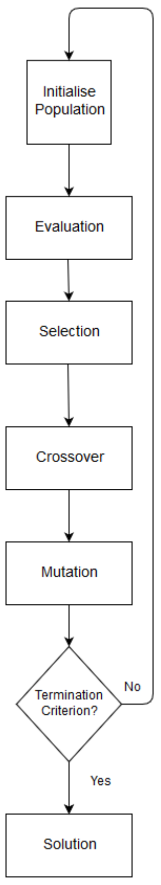


Figure 2.12: Genetic Algorithm Steps

2.5.1.1 Genetic Algorithm

Over the years, engineers have been looking to Nature, seeking to find solutions for computational problems. As an example, by replicating the behaviour of swarms and biological neural networks, Ant-Colony algorithms and Artificial Neural Networks were developed respectively [106]. Thus, in this segment, the focus is in Genetic Algorithm, which is inspired in the Darwinian natural selection.

The basis of Darwin theory of evolution is the natural selection of individuals in a given population. Where the most suited elements, thrive to propagate their biological heredity to new generations. Hence, increasing the population ability to reproduce and survive [107]. Considering this fact, GA's are robust, efficient, optimization processes, which are modelled based on Darwinism. The main steps for implementing genetic algorithm's are shown in Figure 2.12.

Initialize Population

To begin with, a population of N chromosomes, which each individual represent a possible solution to the problem, is created. A chromosome is represented by a vector of independent variables, constituted by genes, each one coding a component of the vector. Even though, binary coding is widely used, one gene may assume any value[108]. Generally, the population is generated randomly and may contain between several hundreds to thousands of possible solutions. Although the size could vary depending on the scope of the problem[109].

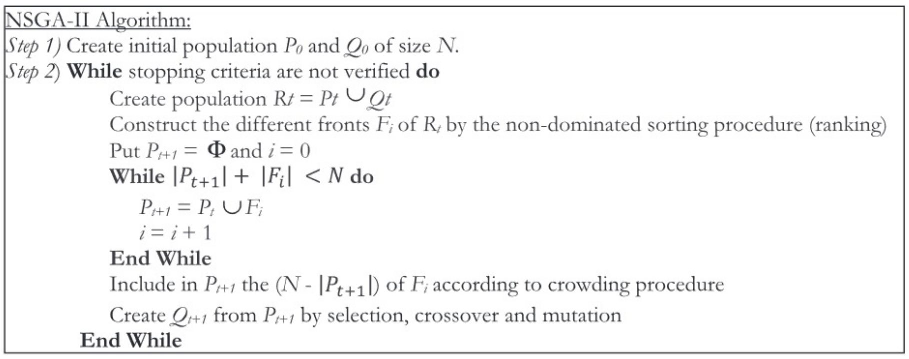
Selection

After arranging an initial population, a selection of the fittest individuals must occur to elect a portion of the existing population to breed a new population. This selection is based on a fitness function, which depends widely with the nature of the problem to be solved. This is the only information that GA's use while searching for possible solutions[110]. To draft the best individuals to enter the mating pool, several selection methods are proposed in literature, though among the most used are:

- **Tournament Selection**- selects the fittest individuals, by holding a tournament among a varying number of competitors. Usual the tournament is held between two competitors, although, the size could be larger, allowing to weak individuals having more chance to be selected. However, the computational cost of the algorithm reduces as the number of competitors increase. The winner of the tournament is the one who has the highest fitness. Afterwards all the winners are comprised in a mating pool, which has higher average fitness than the original population. Thus, improving the fitness of each successive generation[111].

- **Linear Ranking**- The individuals are ranked from 1 to N, with N the number of individuals, based on their fitness values. The weakest individual is ranked 1 and the best individual is ranked N. The selection probability is linearly assign to the individuals according to their rank. With the fittest individuals having higher probability to be selected than the weakest.

- **NSGA-II algorithm**- is an multi-objective algorithm which approximates the Pareto front, based on the non dominance concept, which was introduced by [112].[113] comprehensively explains the algorithm and the steps needed to its implementation, which are shown in Figure 2.13. Firstly, an initial population P_0 is generated and sorted based on a non-domination procedure. Non-domination procedure consists in yield the solutions which are not dominated by other solutions. A child population Q_0 is then created using tournament selection, crossover and mutation operators. For the following t generations, a population $R_t = P_t \cup Q_t$ is generated. A ranking procedure is then applied to return a list of non-dominated fronts. Thereon, a new parent population P_{t+1}, with the N best solutions is generated completing P_{t+1} with the remaining solutions using crowing methods. To finalize, a new child population Q_{t+1} is created from P_{t+1} by selection, crossover and mutation [113].



```
NSGA-II Algorithm:
Step 1) Create initial population $P_0$ and $Q_0$ of size $N$.
Step 2) While stopping criteria are not verified do
            Create population $Rt = Pt \cup Qt$
            Construct the different fronts $F_i$ of $R_t$ by the non-dominated sorting procedure (ranking)
            Put $P_{t+1} = \Phi$ and $i = 0$
            While $|P_{t+1}| + |F_i| < N$ do
                $P_{t+1} = P_t \cup F_i$
                $i = i + 1$
            End While
            Include in $P_{t+1}$ the $(N - |P_{t+1}|)$ of $F_i$ according to crowding procedure
            Create $Q_{t+1}$ from $P_{t+1}$ by selection, crossover and mutation
        End While
```

Figure 2.13: NSGA-II algorithm

Crossover

Once the mating pool is selected, crossover and mutation operators are applied to produce the offspring individuals. Crossover operator is applied to pairs of chromosomes with the purpose of exchanging information between a pair of chromosomes to generate a a child for the new generation[114]. Two of the most used crossover schemes are: single-point crossover and two-point crossover (Figure 2.14). For the one-point crossover a section of the parent vector is selected and the genes after the selected point are swapped, generating two offspring chromosomes. The process is similar for the two-point crossover but, instead of having only one point, which after the genes are swapped. It has two points and the sections between these two points are exchanged.

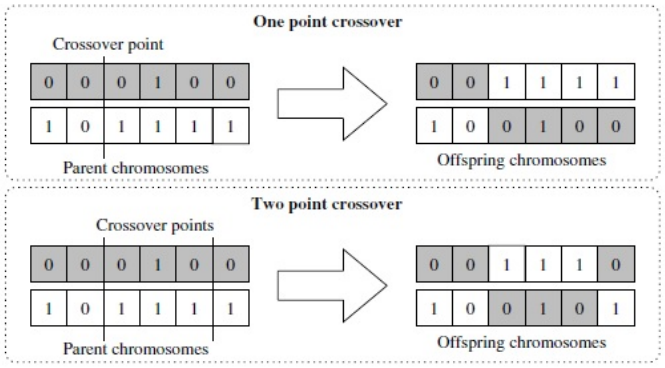


Figure 2.14: Crossover Schemes

Mutation

Mutation operators are applied to single individuals, and the main aim of the operator is to introduce genetic diversity into the population, through introducing random changes to individuals [110]. Some of the more used mutation methods are (Figure 2.15):

- Bit-Flip Mutation- It is used for binary encoded GA's. One bit or more are randomly selected, and their value flips to the complementary value. If the gene as value 1 it changes to 0. Subsequently, if the bit as value 0, it changes to value 1.
- Scramble Mutation - A section of the chromosome is randomly chosen and their index values are shuffled between them, producing a different individual.
- Gaussian Mutation- A selected gene value from the chromosome is replaced by a new Gaussian distributed random value. It is only applied for GA's dealing with integer or floating genes.

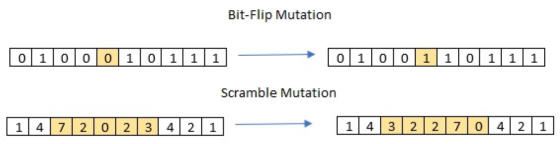


Figure 2.15: Mutation Operators

Termination Criterion

Deciding when the GA shall stop processing, is a vital aspect. Regarding the convergence to the optimal solution, this aspect widely depends on the problem in question and the data provided to the genetic algorithm. A long-established stopping criterion is to interrupt the GA after a predefined number of generations. Furthermore, another widely used stopping criterion is the convergence of the optimization process. After multiple generation's the fitness value tend to converge to a optimum value. And so, if the value stagnates or only a slightly increase is noticed, it means that the algorithm found a reasonable solution. Therefore, a better solution it is not likely to be found in the future, thereby, the algorithm could stop with little to none prejudice to the fitness value.

Regarding the latter approach, if the GA converge too fast to a certain fitness value. It is likely to be stuck in a local optimum, hence, missing the global one. Therefore, compromising the optimization of the fitness function[110].

As previously stated Genetic Algorithm's are extensively used as the way-to-go search heuristic, to find quasi-optimal solutions for maintenance related scheduling problems[113].[108] proposes a multi-objective GA to a continuously monitored system, to determine the optimal degradation level for performing a PM action. It blends Monte Carlo Simulation method to describe the degrading state of the system with GA to optimize maintenance decisions. [86] Proposes GA for maintenance optimization to a system with previously acquired prognostic information. It demonstrated that a threshold based maintenance is not optimal proposing an approach to maintenance decision without thresholds.

[5] introduced a jointly optimization of maintenance schedules and throughput adjustment operations. As in [108], applied a combination of GA and Monte Carlo Simulation.

Regarding the genetic algorithm, instead of a traditional single row chromosome, uses a matrix chromosome with size N x M, with N as the number of machines and M the distinct time-moments. In order to optimize the system reliability of a multi-factory system with age reduction factor, [115], implemented a Genetic algorithm consisting of two inter-connecting parts. One part in control of the job allocation procedure and the other responsible for the production and maintenance scheduling. For validation, the model undergone through three examples. The first demonstrated that the proposed approach is able to maintain the reliability in a predefined acceptable level. The second demonstrated that when it is required an higher reliability, the system react, increasing the number of maintenances. The third revealed that as the age reduction factor increases, maintenance actions increase as well.

As final note, along the present chapter several key concepts of this dissertation, such as Reliability and Maintenance, were explained. Though, the most important was to study which approaches are currently being explored in literature. In order to set a good starting point, to explore new ideas that could improve production systems performance in the future.

Chapter 3

Methodology

In Chapter 2 the main concerns that manufacturing companies face nowadays were presented as well as the main approaches explored to overcome these issues.

Throughout this chapter, the problem that this dissertation is proposed to solve is presented, as well as the proposed methodology as a conceptual solution.

3.1 Problem Definition

One of the main issues of maintenance is to guarantee the availability of production systems. To reach this goal, maintenance departments define their own policies. For each machine to be maintained, usually preventive and corrective maintenance policies are adopted. The resulting maintenance plan can be performed and leads to minimization of maintenance costs. Nevertheless, maintenance planning is often changed due to production constraints or unexpected breakdowns [116].

For instance, when a machine is severely degraded, one could lower the throughput settings of a machine, decelerating the degradation rate, or in other words, decreasing the load and production rate of the machine. By doing so, maintenance actions can be performed in more opportunistic time frames resulting in an increased availability and productivity of the whole system.[5]

Otherwise, dynamically varying the throughput rate of each machine could also be helpful to counter the effect of either expected and unexpected machine downtimes on the total system production. To satisfy production goals, it might be needed to increase the load of the remaining machines to compensate the absence of one or more machines, which are undergoing maintenance activities. Although, as load increases, so does the degradation rate (Figure 3.1[5]).

In this dissertation, changeable throughput capabilities are combined with predictive maintenance techniques, which schedule maintenance activities based on machine continuous monitoring. Thus, it is supposed that a prognostic system keeps continuously evaluating the remaining useful life for each machine. However, RUL is a limit that should not

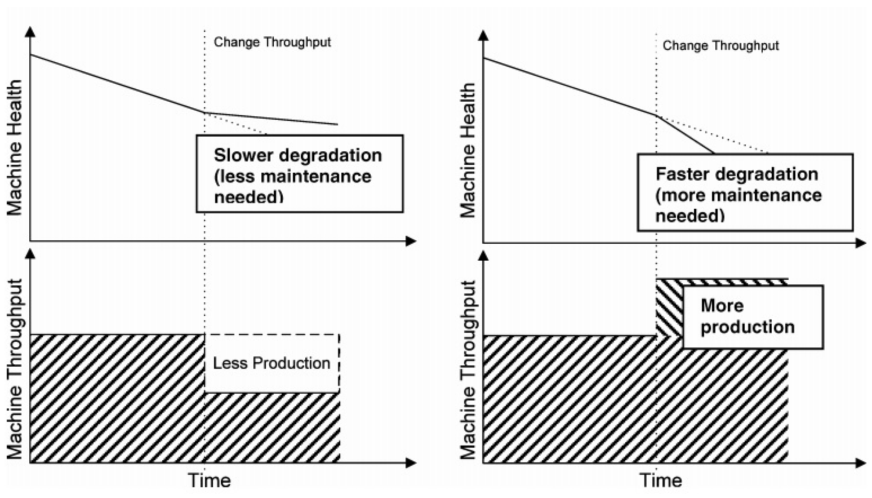


Figure 3.1: Throughput Rate Effects

be crossed so, before the given date, a preventive maintenance action has to be planned. This is where a decision problem arises. Without one machine producing, what should be the load profile of the remaining machines to offset the production shortfall in order to meet short-term production needs?

Therefore, the problem here proposed to solve is to reduce maintenance costs. By exploring the benefit of bringing together predictive maintenance approaches to monitor machine degradation levels, and scheduling preventive maintenance activities, it is possible to 1) avoid unexpected failures in production throughput according to the MPP and 2) dynamically change the machines' load to outweigh the effects of preventive maintenance activities, when clear production goals are pre-defined. Hence, optimizing throughput settings is the key to satisfy production expectations.

3.2 Conceptual Solution

As previously stated, the goal of this dissertation is to study the benefits of a joint approach of predictive maintenance and optimization of load settings to reduce maintance costs and to follow a MPP. For that purpose, machine sensors need to be continuously monitored in order to predict the future behaviour of each asset. Therefore, a prediction model needs to be implemented. To begin, feature engineering techniques must be applied, which, enhance the classifier performance by, creating new features based on the data provided. With the new processed data, the classifier is trained and tested. After the training process, the classifier is able to predict failures. Online sensor data is then acquired and analysed based on the trained model to predict a possible failure in the near future.

After the classifier signalises that an asset is due to fail in the short-term, the prediction model schedules a maintenance activity right before the predicted machine failure, avoiding unscheduled maintenance actions (Figure 3.2).

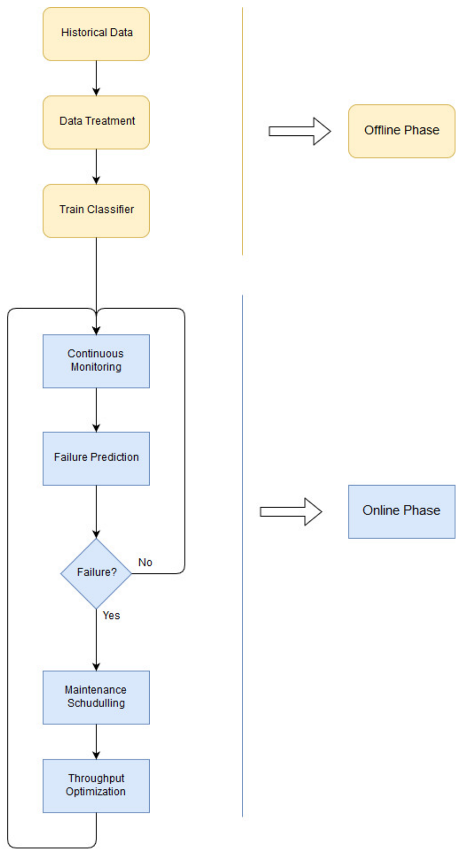


Figure 3.2: Solution Conception Diagram

Therefore, the role of the prediction model is to prevent the occurrence of unexpected failures using a classifier, and to schedule maintenance actions right before a predicted machine failure. Thus, this approach does not optimize when maintenance actions occur. Instead, it focus on scheduling maintenance activities based on the continuously degradation of each machine. It doesn't contemplate any external variables, such as production plan, spare parts inventory or maintenance team availability. To overcome this issue a joint approach of prediction model and an optimization model is proposed.

Thereby, the optimization model is responsible to optimize the production of the whole assembly line. As maintenance actions are held, production rate decays and production targets can not be fulfilled. To comply with the objectives after a maintenance is scheduled, the optimization model is called and generates a set of possible solutions to meet the production demands. Each solution represents a sequence of throughput rate variations to be followed by each machine, to outweigh the effects of production shortfall due to maintenance. Therefore, each solution must be evaluated under a cost function and the best possible solution is selected by the model. However, this process is not straightforward, because, when varying the throughput settings of a given day, machine degradation will decelerate or accelerate. Therefore, a failure could occur in the period under optimization, which would compromise the solution by assuming that a machine is producing, when it is actually down for maintenance.

Thus, without any other mechanism, an optimization method is not able to accurately evaluate each solution. To overcome this issue, a simulation method needs to be implemented, to analyse the future behaviour of the machines under a given load sequence. With this approach, when the a possible solution is generated, the evaluation function runs the simulator to properly evaluate each possible solution. The goal is to implement a feedback loop between the Optimization model and Simulation to constantly find new better solutions to conform with production goals.

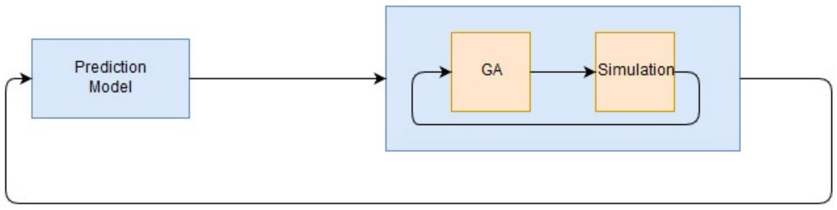


Figure 3.3: System Block Diagram

Figure 3.3 presents the block diagram of the whole system, consisting of a prediction and optimization block, where the prediction block monitors the behaviour of each machine and schedule maintenance based on failure predictions, while the optimization model finds an optimal solution for the load sequence to be applied in the near future. After a solution is generated, the system follows the load settings for each day and continues to monitor the behaviour until a new failure is predicted.

In order to implement this solution a set of tools must be used. The main framework will be built on top of Python, which is one of the most popular languages for scientific computing [117]. It has a high number of well established libraries, which provide tools suited to many tasks. In this project four main libraries will be used:

- Numpy [118] is a Python library which supports a large collection of high-level mathematical functions to operate multi-dimensional arrays.

- pandas[119] data structures and pandas data manipulation routines will be used to extract features from the original dataset.

- Scikit-learn [117] is a machine learning library, which will be used to implement and analyse the prediction classifier.

- DEAP [120] incorporates data structures and tools to implement the most common evolutionary computation techniques. And it will be used to implement the optimization model.

As final note, to evaluate the performance of the proposed solution, two different set-up's will be tested. One making use of the throughput optimization algorithm and one reference test only using the predictive maintenance classifier. To compare both simulations, metrics, such as, OEE and availability will be analysed.

Chapter 4

Implementation

Throughout the present chapter, the implementation process of this project will be detailed and explained.

First, the dataset is analysed, and processed through feature engineering, creating new features to improve the classifier performance.

To make failure predictions, a Gradient Boosting Classifier is implemented. First, data is split into training and test sets. After training, the classifier, based on new measurements, is able to predict the occurrence of failures.

Lastly, a genetic algorithm is used to optimize throughput settings of assets. In order to, minimize the influence of maintenance actions on the overall production plan.

4.1 Data Preparation

Data is the key to solve machine learning problems. A wise choice of which data to use and how to treat it is a crucial step to improve algorithms performances. So, in the current section, the selection process of the dataset to use will be discussed, along with the data preparation phase.

Although it is a growing subject, only a few predictive maintenance datasets were found. The one that gathered more attention was the “Turbofan Engine Degradation Simulation Data Set” published by [121] as a data challenge competition where data from several sensors of a Turbofan engine is provided. But, for the scope of this document, this data was not convenient, since it is focused on air-plane engines. Among factory machines predictive maintenance datasets, one stood out and it was published by [122]. It is rather complete, with measurements of different sensors and it has records of 5 distinct types of data:

- Telemetry Data
- Failures History
- Maintenance History

ht[]

Table 4.1: Telemetry Data

	datetime	machineID	volt	rotate	pressure	vibration
0	2015-01-01 06:00:00	1	176,217853	418,5040782	113,0779355	45,08768576
1	2015-01-01 07:00:00	1	162,8792229	402,7474896	95,46052538	43,41397268
2	2015-01-01 08:00:00	1	170,9899024	527,3498255	75,23790486	34,17884712
3	2015-01-01 09:00:00	1	162,4628333	346,149335	109,2485613	41,12214409
4	2015-01-01 10:00:00	1	157,6100212	435,376873	111,8866482	25,990511

- Errors Log
- Machine Information

The data was collected throughout one year (2015) for one hundred machines. The only exception is the maintenance record, which also records for 2014. Therefore, for the 100 machines the data set contains 876101 hourly-taken telemetry records, totalling 87610 records for each machine. Errors and maintenance files consist of 3920 and 3291 records respectively. Lastly, failure history dataset contains 762 entries, averaging 8 recorded failures for each machine over one year. Each machine has four different components. A single component failure leads to the failure of the machine.

The greatest advantage of using this data set is that the author also implements a predictive maintenance model and achieves great results in fault prediction. Since it is from a trustworthy source, it is complete and has proven good results as a degradation model. Therefore, this was the chosen dataset to be studied in this project.

Telemetry

Telemetry data is one of the most important data to be analysed. It consists of measurements from different sensors annotated with timestamps. As shown in Table 4.1, for each hour, the data of four sensors is provided: vibration, rotation, pressure and voltage.

To better understand the behaviour of each sensor, a simple statistical analysis is depicted in Table 4.2, where mean, standard deviation, minimum and maximum values are compared. This information will be used in the following sections when failure behaviour is analysed.

Table 4.2: Telemetry Description

	machineID	volt	rotate	pressure	vibration
count	876100	876100	876100	876100	876100
mean	50,5	170,7777364	446,6051189	100,858668	40,38500676
std	28,86608652	15,50911421	52,67388604	11,04867939	5,370360562
min	1	97,33360378	138,4320753	51,23710577	14,877054
25%	25,75	160,3049274	412,3057138	93,49818094	36,77729863
50%	50,5	170,6073379	447,5581495	100,4255587	40,2372469
75%	75,25	181,0044934	482,1765999	107,5552314	43,78493829
max	100	255,1247173	695,0209844	185,9519977	76,7910723

Failure History

Each record of Failure History dataset indicates the date and time which a machine had to be replaced due to failure. Figure **??** corresponds to the first five failures for machine ID 1.

Table 4.3: Failure History

	datetime	machineID	failure
0	2015-01-05 06:00:00	1	comp4
1	2015-03-06 06:00:00	1	comp1
2	2015-04-20 06:00:00	1	comp2
3	2015-06-19 06:00:00	1	comp4
4	2015-09-02 06:00:00	1	comp4

Maintenance History

This data set is related to the previous one, providing records of unscheduled maintenances due to failure, as well as, scheduled maintenance due to regular inspection. Table 4.4 presents the first records of the file.

Table 4.4: Maintenance History

	datetime	machineID	comp
0	2014-06-01 06:00:00	1	comp2
1	2014-07-16 06:00:00	1	comp4
2	2014-07-31 06:00:00	1	comp3
3	2014-12-13 06:00:00	1	comp1
4	2015-01-05 06:00:00	1	comp1

Machine Information

The general information of each machine is presented in the Machines dataset, which has 3 different columns, Machine ID, model and age. There are four different models: model 1, model 2, model 3 and model 4. Besides, the oldest machine has an age value equal to 20 and the youngest has age equal to 0 (Table 4.5).

Table 4.5: Machines Information

	machineID	model	age
0	1	model3	18
1	2	model4	7
2	3	model3	8
3	4	model3	7
4	5	model3	2

Errors Log

Registering the occurrence of errors is of high importance. So, all the errors that didn't lead to an immediate failure were documented. Errors' date and time is rounded to the closest hour. Each record consist of a date and time, machine ID and error type. There are five error types: error1, error2, error3, error4 and error5 (Figure 4.6).

Table 4.6: Errors Log

	datetime	machineID	errorID
0	2015-01-03 07:00:00	1	error1
1	2015-01-03 20:00:00	1	error3
2	2015-01-04 06:00:00	1	error5
3	2015-01-10 15:00:00	1	error4
4	2015-01-22 10:00:00	1	error4

4.2 Feature Extraction

Feature Extraction is a fundamental step for Machine Learning (ML) approaches. It is the process of creating new input features based on the data collected, that the learner can better understand[123]. As in [122], different new features were created and brought together to feed the classifier with enhanced data, resulting in better predictions.

Days Since Replacement

In a real-life scenario, as components get older, they become more prone to sudden fail. So, in a predictive maintenance scenario, a valuable feature to extract is the age of each component. However, with the dataset in use, there is no direct way to extract this feature. Hence, by analysing maintenance records it is possible to calculate the time since each component was replaced, resulting in a useful feature which is practically the age of each component

Lag Features

It is conspicuous that time-series forecasting methods make predictions based on past observations. Since past values are likely to be repeated in the future, selecting lag features is a vital step in forecasting problems.

A common method to create lag features is to create a temporal window which comprises N time steps. Several features can be extracted from each window such as: min value, max value, standard deviation, average, etc. The size of the window may vary from problem to problem. Since each dataset has different behaviour and a window of size N might be reasonable to portray the behaviour of the data. Though, if the size of N increases drastically it will probably eliminate any relevant change in the data values.

The data by which lag features are created generally comes annotated with timestamps. For the used data set, telemetry data fits in this constraint. Therefore, two windows were created. One with 3 Hour size to depict short-term changes and other of 24 Hour for a long term history of telemetry data. For each window, the rolling average and standard deviation were calculated.

The addition of lag features is also called the sliding window method. In the 3 Hour window case, as the window is sliding time step by time step, through time, the focus is only on the values within the window. For each new window slide, the average and standard deviation were calculated. The main goal of the sliding window method is by analysing the behaviour of the last window, it eases the prediction of the next time step.

As well as telemetry data, error records come with time stamps, Which make them suitable to be used for lag feature creation, unlike telemetry, where data values are quantitative. Errors ID's have categorical values (Table 4.6). Thus, calculating the average and standard deviation is not applicable. To overcome this issue, error data was reformulated by creating a new column for each error type to count the number of errors for each sliding window step (Table 4.7).

Table 4.7: Error Count

	machineID	datetime	error1count	error2count	error3count	error4count	error5count
7	1	02/01/2015 06:00	0	0	0	0	0
8	1	02/01/2015 09:00	0	0	0	0	0
9	1	02/01/2015 12:00	0	0	0	0	0
10	1	02/01/2015 15:00	0	0	0	0	0
11	1	02/01/2015 18:00	0	0	0	0	0

Machine Features

As Machine features data file only contains merely descriptive information for each machine and is not bound to any time-stamp, they do not need any further treatment. Lastly, every described feature such as telemetry lag feature, days since replacement, errors lag features and machine information, are then merged into a single feature matrix.

Feature Labelling

Lag Features are the classical way that time-series forecasting problems are transformed into supervised learning problems[124]. Therefore, the last step to properly prepare the data to feed the ML classifier is to take each time window prior to a failure and label each time step within the window as "due to fail", while labelling the remaining as "not due to fail".

In the discussed scenario, the objective is to estimate if a given machine will fail within 24 Hours. So, to label each time value 24 Hour prior to a failure, a new failure label was added to the feature matrix. If, for a given 24 Hour window, no failure occurs, all the window segments will have the failure label equal to 'none'. Otherwise, if a failure occurs within the 24 Hour window, all the previous data will be assigned a failure label corresponding to the component which induced the failure. If component 1 was about to fail in 24H, the failure label of the values within 24 H window would be labelled 'comp1'. The same is applicable if it was any other of the remaining components to fail. Four labelled records are presented in Table 4.8. Though, the first records of the whole labelled features dataset is in Appendix A .

Table 4.8: Labelling

	machineID	datetime	voltmean_3h	...	comp4	model	age	failure
0	1	2015-01-02 06:00:00	180,133784	...	170	model3	18	none
1	1	2015-01-02 09:00:00	176,3642932	...	170,125	model3	18	none
2	1	2015-01-02 12:00:00	160,3845679	...	170,25	model3	18	none
3	1	2015-01-02 15:00:00	170,4724608	...	170,375	model3	18	none
4	1	2015-01-02 18:00:00	163,2638057	...	170,5	model3	18	none

4.2.1 Feature Analysis

Data preparation is a crucial step to improve ML algorithms performance. After all features are aggregated, it is important to correlate failure labels and all other features. By doing so, it is possible to highlight which features have more influence on failures. To understand and find correlations, several graphs were plotted. For that, preference was given to scatter plots which proved to be successful finding significant correlations.

Due to its usual importance to understand machine behaviour, telemetry features were first analysed. Using the labelled information previously created, telemetry features were compared. A black dot was assigned to represent the failure label 'none' and different coloured dots to represent the different component labels 'due to fail'. The colours were set as red, green, purple and yellow, for the component 1, component 2, component 3 and component 4, respectively.

Under this circumstances, interesting results were found. As the behaviour of the data set when a failure is due to occur is different from component to component. The following figures represent the failure beahviour of different components, each component failure is assigned with a color.

- Red dots represent component 1 failures
- Green dots represent component 2 failures
- Purple dots represent component 3 failures
- Yellow dots represent component 4 failures

Component 1 Failure

After all telemetry data was analysed to inspect if there is any influence in the 'comp1' label, a high correlation was found between increasing voltage in the last 24 and 3 hours, and component 1 failures. In Figure 4.1, it is possible to observe the mentioned correlation. The features selected were mean voltage and standard deviation of the last 24 hours. However, all the comparisons between voltage features produced similar results.The scatter graph was plotted over the course of 1 month for the 100 machines in the data set. Although several black dots seem to appear in higher voltages. the cluster of red dots is clearly visible.

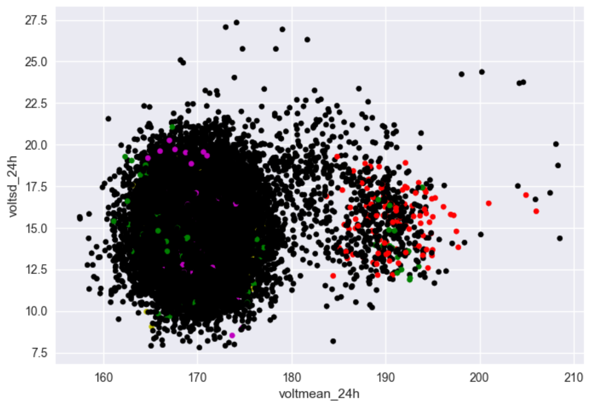


Figure 4.1: Red dots representing component 1 failures due to increased voltage

Component 2 Failure

As in component 1, component 2 'due to fail' labels only have a good correlation with one sensor feature. In this case, as the rotation decreases, component 2 of one machine is more prone to fail(Figure4.2.

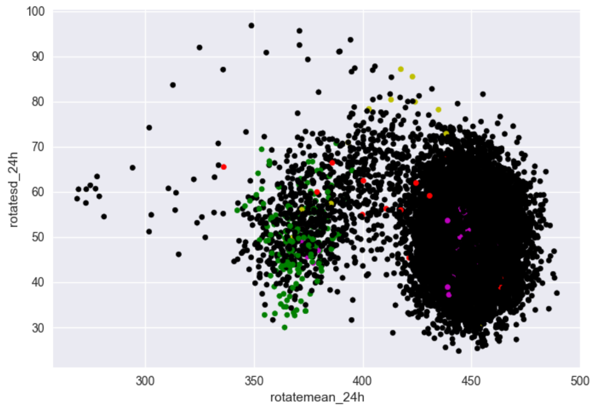


Figure 4.2: Green dots representing component 2 failures due to decreased rotation

Component 3 Failure

As perceivable in Figure 4.3, more purple dots appear in higher pressure levels. Any other sensor type correlated with component 3.

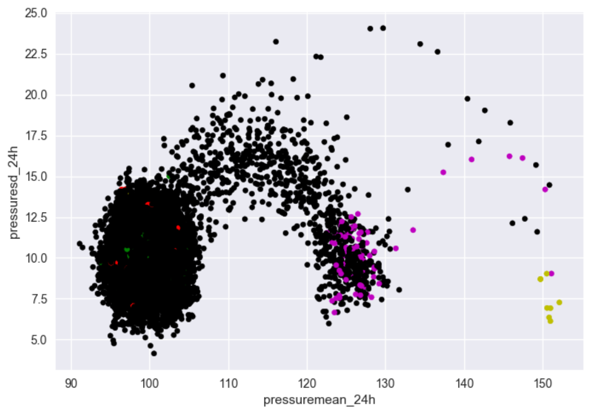


Figure 4.3: Purple dots representing component 3 failures due to increased pressure

Component 4 Failure

Last but not least, and alike other components which had high correlation only with one sensor feature, component 4 is correlated with the remaining sensor. As vibration increases so does the number of component 4 failures (Figure 4.4).

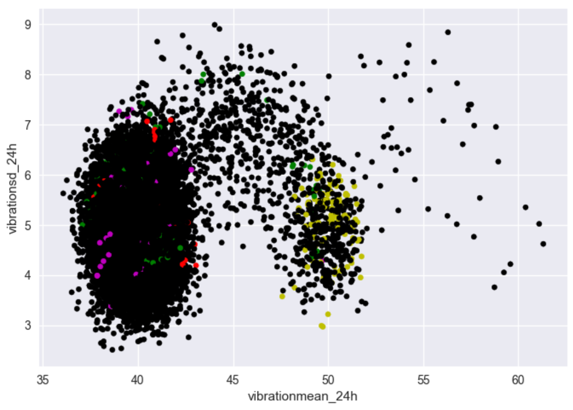


Figure 4.4: Yellow dots representing component 4 failures due to increased vibration

Other non-telemetry features were analysed, such as components and machine ages. Though, only the number of errors within the 24 Hour window time proved to have a good correlation with component failures. Figure 4.5 depicts the behaviour of failure in component 4 (red vertical line), which resulted due to an increase in vibration measurements.

Though, it is noticeable that right before the failure 3 non-fatal errors occurred. The error ID's from left to right are: error type 1, error type 3 and error type 5.

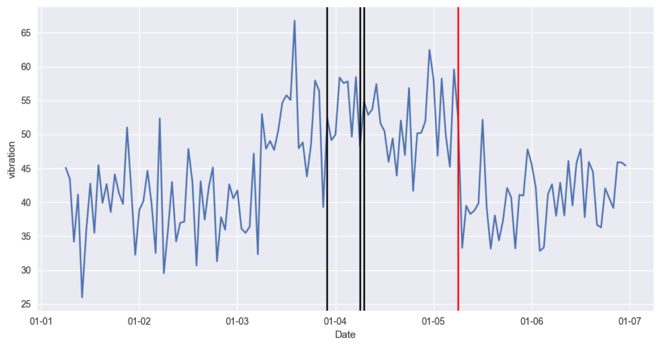


Figure 4.5: Component 4 failure

Afterwards, since telemetry and errors' features proved to have good correlations with the occurrence of failures, and the correlation of telemetry and failures was not perfect, in order to find a better correlation, errors and telemetry features were plotted together. The resulting plots proved that different error types and different sensor measurements when combined generate great results. As in telemetry, different error types influence different component failures.

Component 1 Failure

As previously shown in Figure 4.1, component 1 failures correlate well with the increase in voltage values. Though, if comparing voltage features and errors occurrence, much better correlations were found. As shown in Figure 4.6 if error type one occurred in the last 24 Hours and voltages values are high, it is well likely that component 1 of a given machine will fail.

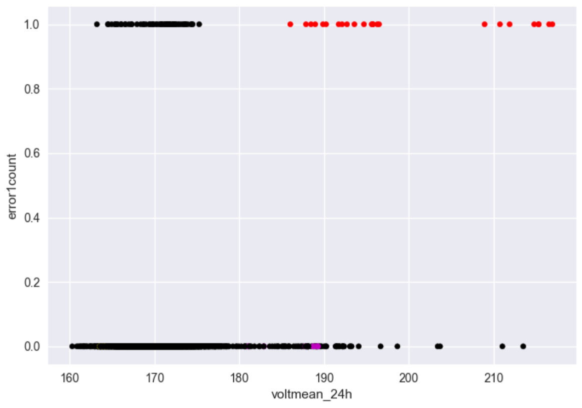


Figure 4.6: Comp1 failures due to increased voltage and errors type 1

Component 2 Failure

Unlike component 1, where, only one error type influence the failure of the component, component 2 failures are influenced by two error types, error 2 and error 3(Figure 4.7).

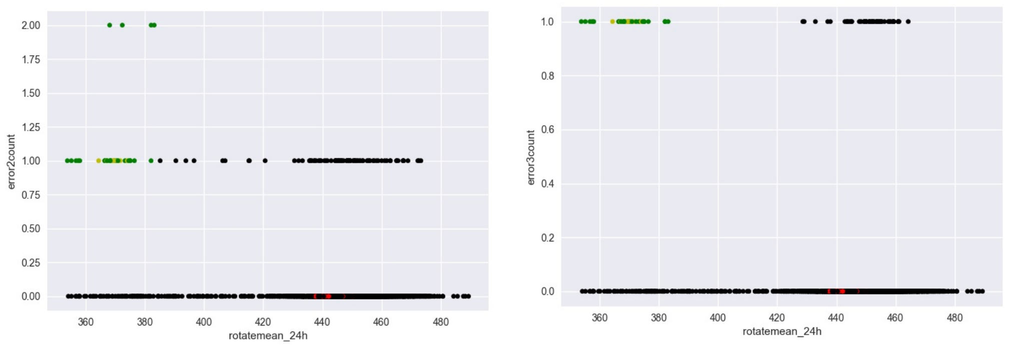


Figure 4.7: Comp2 failures due to decreased rotation and errors type 2 and 3

Component 3 and 4 Failures

As well as in component 1 and 2, components 3 and 4 failures are influenced by one and two errors types respectively. If an error type 4 occurs and pressure values are high component 3 will likely fail in the next 24 hours (Figure 4.8)

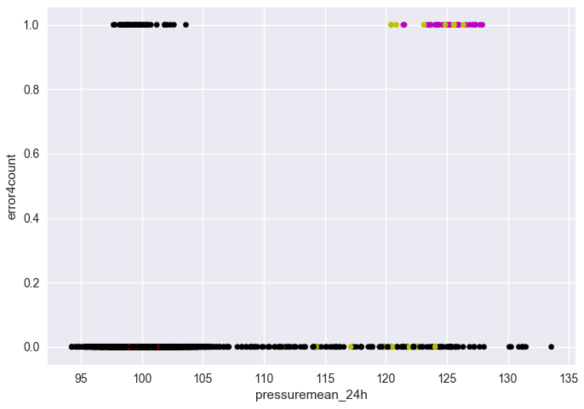


Figure 4.8: Comp3 failures due to increased pressure and errors type 4

Otherwise if Vibration is high and one failure of either type 5 or 3 occur, component 4 will surly fail in the next 24 Hours(Figure 4.9).

4.3 Prediction Model

Based on historical data features and labelling indicating if a failure occurs or not, the role of the classifier is to automatically produce a prediction rule. When unlabelled information

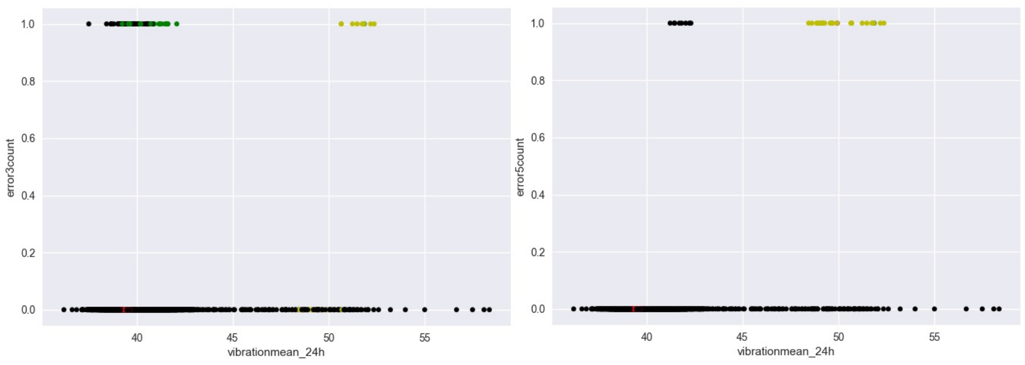


Figure 4.9: Comp4 failures due to increased vibration and errors type 5 and 3

is given, the classifier will attempt to predict if the data represents a failure or not. The goal is to generate a rule that most accurately predicts the state of new measurements.

A comprehensive review of several degradation models has been made in chapter 2. As a conclusion, in literature, Artificial Neural Networks and Bayesian related approaches had more successful and promising implementations. Among the studied papers, the one that drew more attention was an implementation proposed by [62]. The author explores the usage of Dynamic Bayesian Networks for a predictive maintenance scenario. This solution is worth mentioning since it is a generic approach which takes into account different operational issues such as, dynamic environment and load variations, which would perfectly suit the needs of this project. Even though more interesting solutions are proposed in literature, foor this project, the chosen approach was to implement a Gradient Boosting Classifier, which was the approach selected by [122]. Albeit, in predictive maintenance literature had not been extensively explored as other approaches, it was a natural decision for this project since there are confirmed good results using the dataset in study.

Boosting algorithms emerged from questioning the possibility of converting a weak learner into a strong learner. In ML a weak learner is defined to perform just slightly better than random chance, whereas, a strong learner, should be able to yield a correct classification rate close to 100%. Since it is always much easier to construct a weak learner than a strong one, the answer of this problem is of high practical importance. Therefore, the concept of Boosting Algorithm is that any weak learner can be potentially iteractively improved (boosted) to become also a strong learner[125].

As an example, when building a classifier to detect e-mail spam, achieving a highly accurate prediction rule might be difficult. Although, if several "not so accurate" (weak learner) rules are combined, a similar end-result might be achieved with a lower level of complexity. An example of "not so accurate" rule could be: "If the phrase 'buy now' appears, then predict is spam.". To yield a good prediction, the algorithm calls the weak learner repeatedly, feeding it with different subsets of the training set. And, each time the algorithm is called a new weak prediction rule is generated. Lastly, the better performing solutions are given a higher weight and combined into a single and more accurate prediction rule[126].

Gradient Boosting Algorithm

Gradient Boosting Algorithm is a boosting method, with a learning procedure that consecutively fits new models to provide a more accurate estimate of the response variable. The principal idea behind this algorithm is to construct new base-learners to be maximal correlated with the negative gradient of the loss function, associated with the whole system. The loss function is an arbitrary choice, thus, least square regression is commonly used[127]. Using a training sample $\{y_i, x_i\}_1^N$ of known x and y, where x = $\{x_1, ..., x_n\}$ refers to the explanatory input variables and y to the correspondent labels of the response variable.

The goal is to obtain an estimate $\hat{F}$, in form of the weighted sum of base-leaner functions $h_j(x)$, of the unknown function dependence x $\rightarrow$ y, that minimizes the expected value of some specified Loss function L(y,f(x)) [128].

$$F^* = \underset{a}{argmin} E_x[E_y(L(y, f(x)))|x] \tag{4.1}$$

Where y is the response variable, f(x) is the true functional dependence, $E_y(L(y, f(x))$ the expected y losses and $E_x[E_y(L(y, f(x)))|x]$ the expectation over the whole dataset.

The fundamental idea of gradient boosting is to fit the base-learner a negative gradient vector of the loss function evaluated in the previous iteration. Hence, in every boosting iteration the base-learner is directly fitting the error made in previous iterations. Therefore, boosting the performance of a simple base-learner by iteratively shifting the focus towards problematic observations that are difficult to predict. [125].

For this project, a pre-implemented Gradient Boosting Algorithm by [117] was used. It was developed as part of python package, scikit learn, that supports a straightforward implementation of the algorithm, which also provides built-in evaluation metrics.

An important step to implement any Machine Learning Classifier is to partition the data into training and test sets. When dealing with time-stamped data, lagging features are often used. Partitioning data must be done carefully to avoid overestimating the performance of the algorithm. Since records in the same time-window likely have the same label and similar feature value, partitioning data in the same window would give an unfair advantage to the algorithm when predicting a test set record that shares its time-window with a training set record [122].

To minimize the number of shared intervals between training and test sets, it is wise to apply a time dependent record splitting strategy. A date point is selected and every record prior to that date is assigned as training set, and the records after as test set. However, one shared "border" still remains. To overcome this issue, a new split has to be made in order to ignore 24 Hours worth of records between training and test sets.

In order to evaluate the performance of the algorithm, three models with different training and test sets threshold dates were created. To compare the performance for each split, several metrics, such as, confusion matrix (Tables 4.9,4.10,4.11) were calculated.

Table 4.9: Confusion Matrix Split 1

		True Labels				
		'None'	'Comp1	'Comp2'	'Comp3'	'Comp4'
Predicted Labels	'None'	120285	21	0	4	3
	'Comp1'	18	515	2	5	2
	'Comp2'	0	1	867	0	1
	'Comp3'	12	0	2	373	1
	'Comp4'	2	1	6	0	498

Table 4.10: Confusion Matrix Split 2

		True Labels				
		'None'	'Comp1	'Comp2'	'Comp3'	'Comp4'
Predicted Labels	'None'	95957	13	0	4	3
	'Comp1'	19	400	1	1	1
	'Comp2'	0	1	707	0	0
	'Comp3'	12	0	2	291	1
	'Comp4'	2	2	4	0	392

Table 4.11: Confusion Matrix Split 3

		True Labels				
		'None'	'Comp1	'Comp2'	'Comp3'	'Comp4'
Predicted Labels	'None'	72415	7	0	4	3
	'Comp1'	17	299	1	1	2
	'Comp2'	0	1	555	0	0
	'Comp3'	11	0	0	212	1
	'Comp4'	2	0	3	0	275

Confusion Matrices are used as a method to describe the performance of a classifier. Typically they are only frequent in supervised learning algorithms where labelled data is provided. Essentially, it compares the predicted labels with true labels, presenting the number of correct and incorrect predictions.

As an example in Table 4.10, the labels in the X axis represent the true labels, and Y axis the predicted labels. Therefore, the diagonal values depict the number of times the algorithm was able to correctly classify the records of a given label. For example, in the same table, the algorithm successfully predicted 400 out of 416 'comp1' records. However, 13 'comp1' records were predicted as 'none', 1 as 'comp2' and 2 as 'comp4'.

As clearly noticed, the number of records classified as 'none' is overwhelmingly superior to the records of the remaining labels. This is mainly due to the nature of the problem. As, comparing to the normal operation time, machine failures rarely occur. Thus it causes an imbalance in label distribution, which might be an issue to access the performance of the algorithm. As algorithms tend to classify majority class examples better at the expense of minority class examples as the total misclassification error is much improved when majority class is labelled correctly. Due to the class imbalance problem, it is important to look at evaluation metrics other than accuracy [122] to fully understand how the classifier is actually performing. Even though the accuracy remains high, this could lead to low recall rates.

One of the most important metrics in predictive maintenance is the model recall. It represents the number of actual failures that were predicted. This metric is valuable since it is influenced by the occurrence of false negative predictions. False negative predictions are failures that the model didn't predict. Thereafter, recall rates for each model were calculated and presented in Table 4.12, testifying the good performance of the algorithm to predict failures, as the recall values for every category were well above 90 %.

Table 4.12: Recall Rates

	'none'	'comp1'	'comp2'	'comp3'	'comp4'
Recall for first split	0.9998	0.9501	0.9977	0.9613	0.9822
Recall for second split	0.9998	0.9479	0.9986	0.9510	0.9800
Recall for third split	0.9998	0.9344	0.9982	0.9464	0.9821

Supported by these results, split 1 was chosen to be the training set for the real-time model.

4.4 On-line Simulation

After training, the classifier is able to predict short-term failures based on real time machine monitoring. To simulate the real-time behaviour of each machine, unlabelled data from the dataset could be used to continuously feed the classifier. However, this approach would be quite restrictive, since it wouldn't allow any variations to the original dataset. Therefore to mimic the real-time behaviour of the system, two windows were selected. One corresponding to the normal machine operation, where no errors occur and sensor measurements are in normal values. This window comprises pf eight measurements taken each 3 Hours, representing one day worth of data. The other window represents the failure mode of a given machine. It starts right before sensor measurements start to increase and ends when the failure occurs. These windows were selected from a random failure that took place in split 2 (Figure 4.10).

For the scope of this dissertation, taking into account multicomponent machines would greatly increase the complexity of the simulation implementation with few influence in the final conclusions. Therefore, due to demonstrative purposes, this project will only be based on component 4 failures which are mostly influenced by vibration values.

One big advantage of having 2 different types of windows is the possibility of easily change the MTTF of each machine by varying the number of "normal-state" windows each

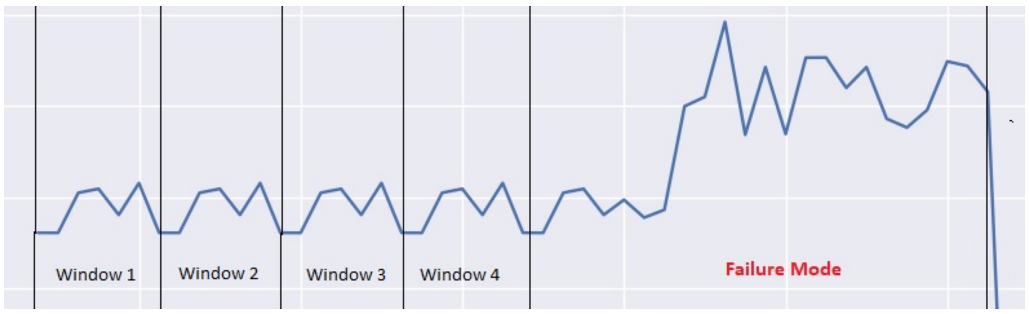


Figure 4.10: Window slicing

machine has through its life cycle before entering in the "failure-mode" window. Therefore, to represent an older machine with lower MTTF, it would only be needed to lower the number of slices, whereas, for a newer machine with higher MTTF, rhe number of slices would have to increase. Figure 4.11 depicts the normal degradation path of two machines, where one has higher MTTF (Machine 1) and Machine 2 has a lower MTTF.

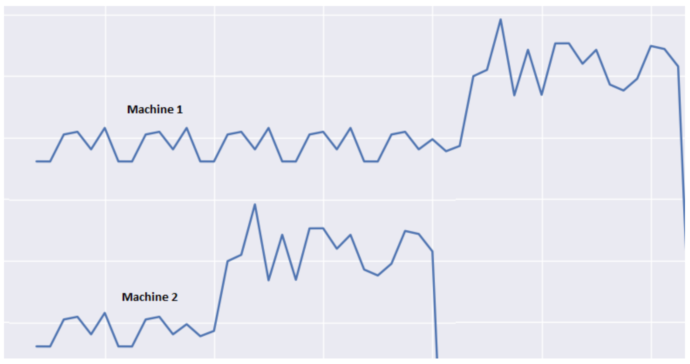


Figure 4.11: MTTF Comparison

Maintenance

As new records are sampled, all the data is fed to the classifier which predicts if a failure is due to occur. If a failure is predicted, a maintenance action must be scheduled. Thereby, the prediction model must write into a maintenance file the date and time which the machine needs to be maintained. The system will then continuously sample new records until the maintenance date. When under maintenance each machine will be down for a given time period, where it would not produce any part.

Load Settings

As previously established, after a maintenance action is scheduled a load optimization algorithm is called which is able to change the throughput rate of each machine. Five different load settings were implemented by varying the sampling rate of the simulation:

- Load 0 - Represents the normal behaviour, and the records are sampled normally, one by one.
- Load 1: With this setting , records are sampled 1.5 times faster, and so, MTTF is 1.5 times lower and the production rate 1.5 times higher.
- Load 2: Sampling rate is 2 times faster, resulting in two times more production and each machine will fail twice as often.
- Load -2: Contrary to Load 2, it produces two times less parts than the normal setting, resulting in a higher MTTF.
- Load -1: It fails 1.5 times less often than the normal setting and produces 1.5 times less.

Figure 4.12 provides a good representation of each load setting, and how the MTTF is influenced by it variation.

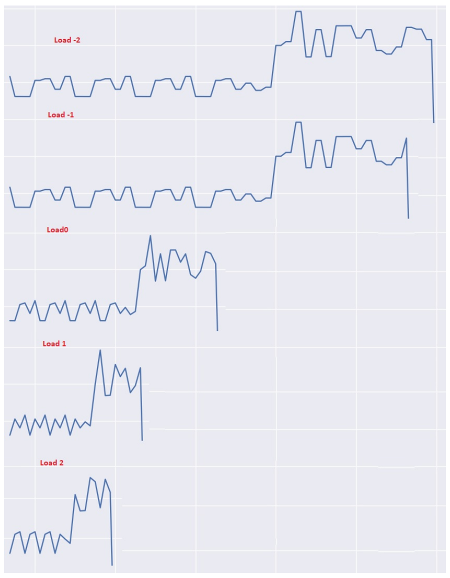


Figure 4.12: Load Settings

4.5 Optimization

As stated in chapter 2, several models are being studied in literature as maintenance optimization approaches. However, Genetic Algorithms (GA) are regarded to perform better than other meta-heuristics yielding better results, closer to the global minimum[129].

In this dissertation, GA is utilized to optimize the throughput rate of each machine after a failure is predicted. Therefore, after the classifier signalizes that one machine is due to fail, the optimization model is called in order to prevent any production shortages in the near future.

For this purpose, production targets are weekly based. Each week the whole system must produce a given number of parts to comply with the objectives. Therefore, the algorithm must only optimize the throughput settings of the week that it was called. Since weeks have independent production targets, after each week, all machines are set to normal load operation, until other maintenance is scheduled and the optimization model is called again.

The first step to implement a GA-based approach is to generate a random population of individuals. The size of the initial population could be chosen by consecutively testing different values and analysing which ones performed better with the problem to be solved [130]. By comparing with similar works in literature and taking into account computational performance, population size was set to be 50 individuals [5].

For a system having N machines, whose load settings could change in M time moments with M equal to the number of time slots left until the end of the week, each individual is represented by a 1D array of size NxM (Figure 4.13), where each element $L_{i,j}$ corresponds to the load setting of machine i at time slot j. Each element may assume five different values: -2,-1,0,1,2. Where negative values correspond to a decrease in throughput rate, zero is the normal state and positive values an increase in throughput rate.

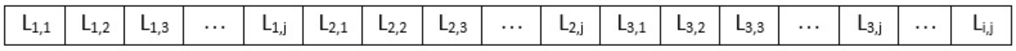


Figure 4.13: Chromosome Structure

Fitness Function

After the population is initialized and before selecting individuals to enter the mating pool, the model must evaluate the fitness of each individual.

The first fitness function created for this project only evaluated chromosomes based on the difference between the week production target and parts produced under the individual load settings. Though, with this approach, the algorithm would always return as best solution the one that produced more parts which was always to increase the throughput rate of each machine to maximum. However, as the throughput rate increases so does the degradation rate, promoting more failures to occur. In many cases increasing the throughput rate would not be necessarily the best solution, since the total production at the end of the week would largely surpass the target. In a real-life scenario, machine failures are to be avoided, since it represents machine unreliability leading to an increase in maintenance costs. To further avoid favouring solutions that surpass the target, a constraint was added, that the difference between the target and the produced parts could not go lower than 0.

To get a more realistic trade-off between production and degradation rate, two penalties were added to the fitness function. One, P_{in}, to account for new failures provoked by changing load settings in the week being optimized. And one to avoid that variation in throughput rate in the present week would affect the production of the following week. Therefore, P_{out} represents the number of failures in the first two days of the following week.

Since the model could not possibly count how the system would behave under a given load sequence to count the number of failures, a simulation mechanism was implemented. After an individual is created the role of the simulation is to predict how the system would behave under the load sequence given by the individual. To do so, it makes use of all the past observations, and with a similar procedure as the on-line simulation, it reproduces the behaviour of the system allowing to count the number of failures.

The last terms added to the fitness function were a standard deviation penalty, which prevents harsh variations between time slots, and a penalty to count the number of changes in load settings in each solution, which promotes a more homogeneous solution, decreasing the number of times the load needs to be changed

$$F = \left(Target - \sum_{i}^{N}\sum_{j}^{M} X_i \cdot L_{ij}\right) + \sum_{i}^{M} P_{in}\cdot C_{in_i} + \sum_{i}^{M} P_{out}\cdot C_{out_i} + \sum_{i}^{M} P_{sd}\cdot C_{sd_i} + \sum_{i}^{M} P_{ch}\cdot C_{ch_i} + P_{np} \tag{4.2}$$

Equation 4.2 represents the fitness function implemented. The first argument is the difference between the Target and the total week production represented by the product of the normal production rate of machine i, X_i, and the throughput rate of machine i in time slot j, L_{ij}. Second and third arguments represent the penalty due to failures in and out of the week. P_{np} was added as penalty to not comply with the target. P_{in} and P_{out} are the fixed penalty for each failure. Whereas C_{out_i} and C_{in_i}, are respectively, the number of failures of machine i in the present week and the following. Lastly, P_{sd} and P_{ch} are correspondingly the fixed penalties for standard deviation variation and per number of different load settings. C_{sd_i} is the standard deviation of the chromosome split correspondent to machine i and C_{ch_i} is the number of changes in the chromosome split correspondent to machine i.

Genetic Operations

After the fitness of each individual is calculated, a set of operations to promote genetic diversity and convergence to the global optimum must be applied. Thereby, each successive generation would have better fitted individuals. After a defined number of generations is reached, the algorithm stops and yields the fittest individual of the current generation. The number of generations is the stopping criterium of the algorithm. In literature the number is usually around 100[6], though depending on the nature of the problem it could be higher or lower. Due to computational constraints, in this dissertation this value was decided to be 30, since it provided good solutions and as in the example of Figure 4.14

before 30 generations the fitness started to converge. The trade-off here is to benefit computational efficiency over possibly be stuck in a local optimum.

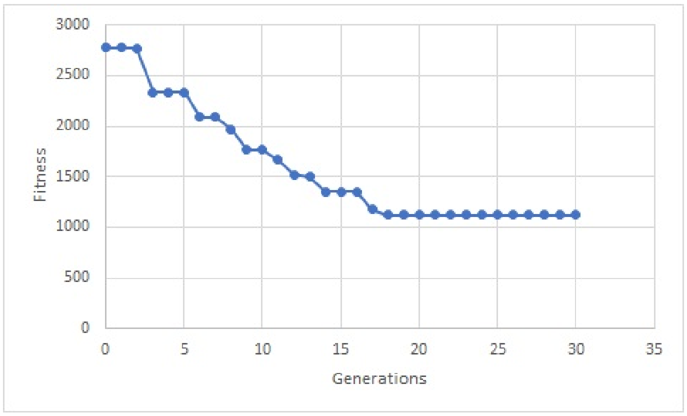


Figure 4.14: GA Convergence

Genetic Operations were implemented using Distributed Evolutionary Algorithms in Python (DEAP) framework[120]. This framework provides pre-implemented algorithms and genetic operators, which simplifies the creation of new models. The major drawbacks are the computational cost of the framework and the inflexibility to constrain non admissible solutions. Since in this project non admissible solutions are easily constrained, crossover, mutation and selection were implemented using this framework.

Selection

The selection phase determines which individuals are picked for the mating pool. The main principle of selection is the better an individual is, the higher is it chance to be selected and propagate its genetic information to the next generations.

Selection introduces the influence of the fitness function to genetic algorithm optimization process by comparing the "goodness" of each individual and selecting the fittest. Thus, selection can't be solely a choice of the best individuals, since the best individuals of a given generation may not be close to the global optimum. Therefore, in order to ensure that the information carried by presumed unfit individuals is not prematurely lost, selection methods must allow a percentage of unfit individuals to enter the mating pool[131].

Many selection methods are proposed in literature, though many point out that tournament selection has the best performance and it is computationally efficient [131] [132]. Tournament selection elects N individuals to be part in a tournament, where the fittest passes through. Generally the size N of the tournament is 2, although this number could be higher. Larger tournament sizes lead to loss of diversity in the population, since "worst" individuals have less chance to reproduce, contributing with a faster convergence, with higher probability of the model be stuck in local optimum. Due to the above reasons, and to ensure diversity in each population, tournament size was set to 2.

Crossover

Chromosome crossover is a method of exchanging genetic information between two chromosomes. The crossover methods analysed in Chapter 2 were 1-point crossover and 2-point crossover. Single point crossover promotes less diversified offspring's, since only one section of each individual is exchanged. Whereas two-point crossover promotes more diversity when generating new individuals. Since each chromosome consists of N independent blocks, being N the number of machines, two-point crossover was chosen to allow information of more than one block to be exchanged, generating more diverse solutions among each block.

Regarding crossover, the other decision to be made is to determine the ratio of next generation individuals that are to br generated through crossover. Crossover rate is an arbitrary value that highly depends on the nature of the problem. One of the best ways to determine the more efficient rate is to iteratively test several rates and determine which performed better.

Due to computational constraints, testing several rates was not doable. Therefore, crossover rate was chosen based on similar problems proposed in literature where crossover rate was always around 70 %[5].

Mutation

In crossover operation, the main objective is to lead a population to converge to a good solution, although on the opposite side, mutation operator is meant to avoid this convergence, adding genetic diversity to prevent convergence to a local optimums.

As in crossover, mutation has a rate representing the percentage of individuals that will suffer mutation before the next generation. Either mutation rate and mutation methods were chosen solely based on literature proposes. Random mutation was selected as operator, and it randomly selects different cells and swap their indexs [6]. As in [5] mutation rate was set to 5 %.

Chapter 5

Experimental Validation

5.1 Experimental Setup

To evaluate the implemented model, a manufacturing system consisting of 3 machines producing in parallel, whose layout is shown in Figure 5.1 was tested.

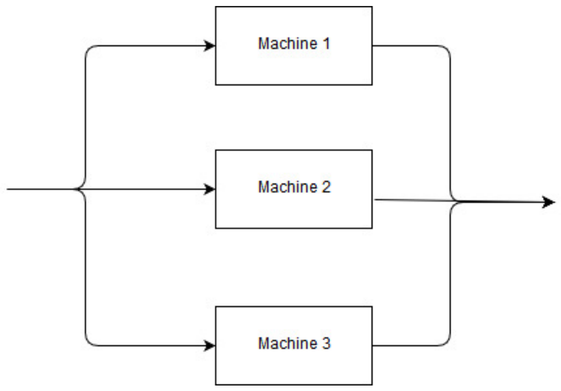


Figure 5.1: System Layout

As it is a parallel layout, the whole system production is equal to the sum of the parts produced by each machine. The experiment simulates the system for 56 days, resulting in 8 production weeks. Each week the manufacturing system should comply with a defined target of 2400 parts/week. When all machines are producing at normal rate, the total parts produced by the system is equal to 2520, having a surplus of 120 parts. However, when a machine is under maintenance the production of that machine is 0 parts, decreasing the production of the whole system. Maintenance actions are performed by following a maintenance schedule that could be updated by an expert or will be automatically updated by the predictive maintenance model, when a failure is due to occur. To represent a possible real-life scenario where the performance of different machines in the shop-floor is not homogeneous, older machines working side by side with newer and better performing machines is implemented. Different values for each machine repair time, production rate and MTBF were defined, as shown in Table 5.1.

Table 5.1: System Description

Variable	Value
Overall time period	56 days
Number of machines	3
Repair Time M1	1 Day and 21 Hours
Repair Time M2	3 Days and 3 Hours
Repair Time M3	1 Day and 15 Hours
MTBF M1	28 Days
MTBF M2	32 Days
MTBF M3	32 Days
Production Rate M1	5 units/Hour
Production Rate M2	3.3 units/Hour
Production Rate M3	8.3 units/Hour

As an example, machine 2 is the least efficient machine of the system, producing 3,33 parts/Hour with repair time of 3 Days and 3 Hours, and MTBF of 32 Days. Whereas Machine 3 is the most efficient, with the highest production rate of 8,33 parts/Hour, with the lowest repair time of 1 day and 15Hours and with MTBF of 32 Days. These variations in machine performance promote a more realistic example, as well as allow to analyse how the optimization model would behave under different constraints.

Lastly, as the model objective is to minimize the influence of maintenance actions, throughput could vary in 5 different settings as previously stated and could only be changed one time each day. If a maintenance is scheduled, the throughput rate could not be changed, remaining the same load setting as the time slot which the maintenance was scheduled.

5.2 Results and Analysis

To fully understand the performance of the proposed solution, two different tests were made, one with the optimization algorithm and one without, and with every machine at normal throughput rate. Figures 5.2 and 5.3 represent the vibration sensor measurements for the algorithm test and the normal test respectively, providing a general overview of how each machine behaves over time under the different tests. In both Figures weeks are delimited by vertical white lines. When a machine has high vibration values it is entering the failure mode, whilst when the sensor value is 0, it signifies that a maintenance activity is being performed. As presented in Table 5.1, the total runtime of each test was 56. Which extended from 01-01-2016 to 26-02-2016.

The reference test, without the optimization algorithm, only ran the simulation with the predictive maintenance model. Therefore, each time a failure was predicted, a maintenance action was scheduled and then performed. The throughput rate of each machine for this test was always at normal state. Figure 5.2 displays the behaviour of each machine throughout the test. As it is noticeable, 6 failures were predicted. However, since machine

2 last predicted failure occurred in the last simulation date, only 5 maintenance actions were performed.

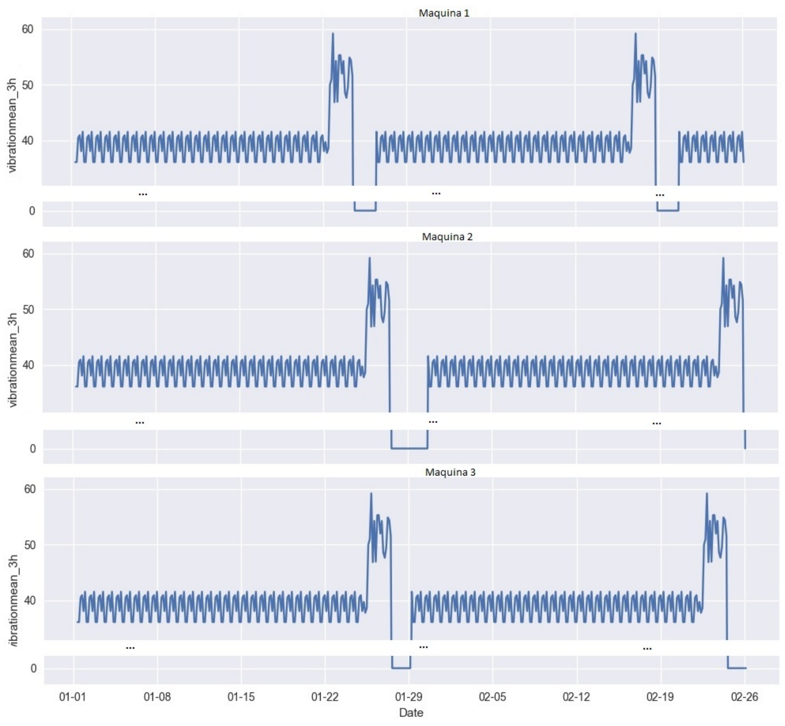


Figure 5.2: System behaviour without Optimization model

The second test was made using all the implemented features, predictive maintenance model and throughput optimization model. The system behaviour throughout the test is presented in Figure 5.3.

As well as in the reference test, after each failure predicted by the classifier a maintenance action is scheduled. Though, in this test as the maintenance action is scheduled, the throughput optimization model is also called. In Figure 5.3, the vertical red lines represent the time the optimization model was called. As it is easily perceived, the model was called in four different occasions. First due to a machine 1 failure, then due to machine 2, the third failure was due to machine 3 and lastly, due to machine 1 again. Since the model optimizes the load settings for the whole week, if more than one failure occur in the same week, the model is only called for the first one.

In some cases in Figure 5.3 is possible to notice the effects in the sensor behaviour caused by the alteration of throughput settings. For example, after the first failure it is noticeable a decrease in the sampling frequency in machine 2 sensor, induced by a decrease in the throughput rate for that machine. Whereas in machine 3 it is noticeable an increased sampling frequency representing a higher throughput rate.

In the second test five failures were predicted by the classifier, and the corresponding five maintenance actions were performed.

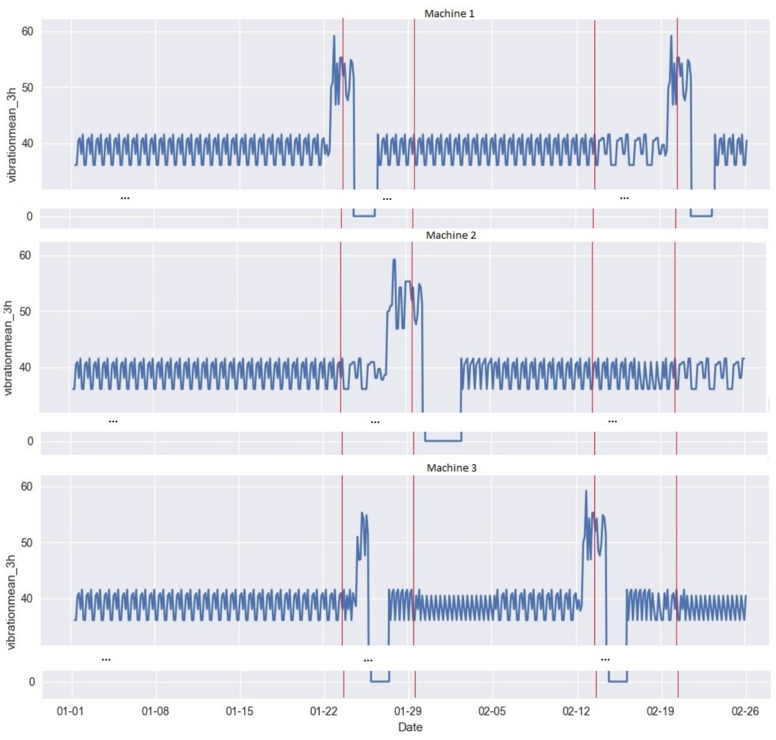


Figure 5.3: System behaviour with Optimization model

To get a better perception of how the throughput rates evolved over time, in Figure 5.4 the step plots of load variation over time for each machine are displayed. The yellow area represents the period between the scheduling and the actual maintenance, and the red line corresponds to the period a given machine is under maintenance. Therefore, load variations under these periods are meaningless, since while in the "yellow period" the actual throughput rate would be equal to the time period immediate before maintenance is scheduled. As in Figure 5.3 and 5.2, vertical lines limit each week.

One of the best examples, to analyse the algorithm behaviour is the period between 29/01/2016 and 05/02/2016. At the beginning of that period, machine2 had to perform a maintenance activity, represented by the beginning of the yellow section, while in that period machine 2 is in a state " due to maintenance", it cannot change it throughput. After that state, a maintenance action is performed, marked by the red section, where the asset is not producing. To overcome the lack of production, the algorithm, in order to meet with production requirements, must offset the deficiency in production by increasing the throughput of the system.

Table 5.2: Production With Algorithm

	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8
Production With Algorithm	2520	2520	2520	2420	3365	2520	2555	2960
Balance	+120	+120	+120	+20	+965	+120	+155	+560

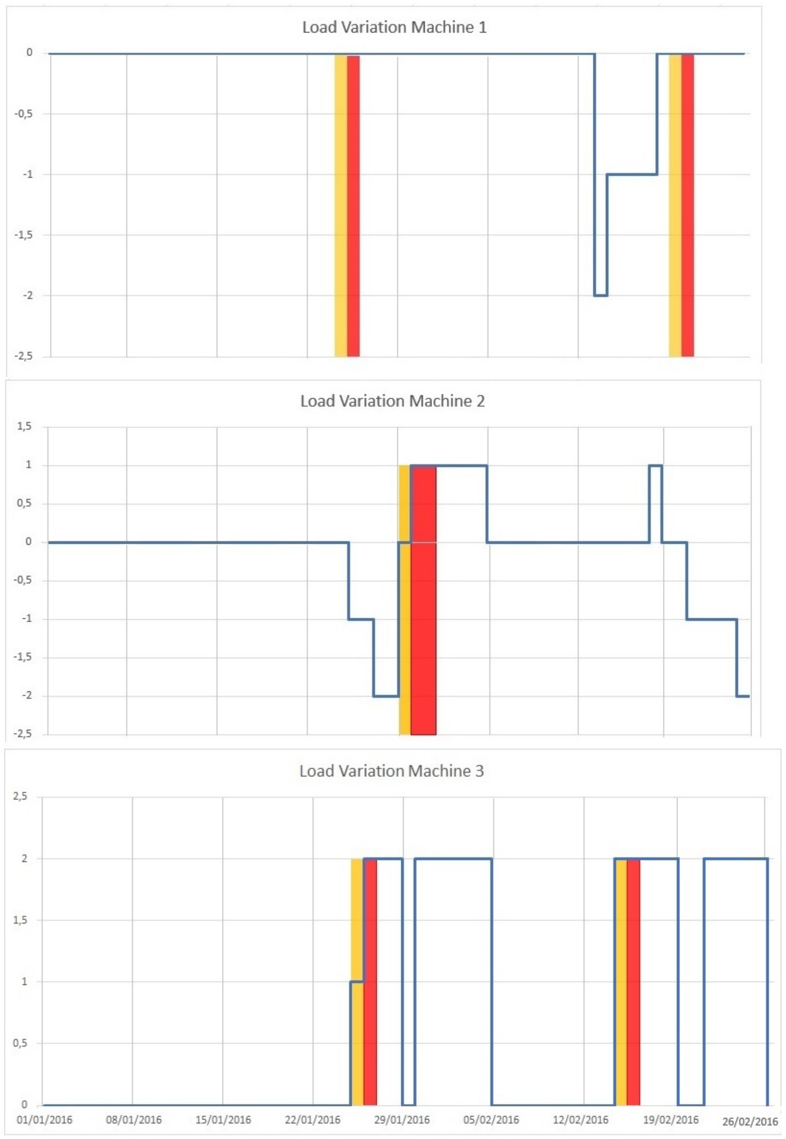


Figure 5.4: Throughput Rate Variation

In figure 5.4 it is possible to observe that the algorithm followed the assumption that needed more parts being produced, and adjusted the throughput rate to meet the target. In that period machine 3 increased the throughput rate, represented in Figure 5.4 by the step from setting 0 to load setting 2 (maximum rate). In the same week the throughput of machine 2 also increase to load setting 1 (1.5 times production rate). This setting will only be applied after the maintenance action is concluded.

Even though penalties were added to not reward the system if the target was surpassed, the production of that week was 965 parts more than the required target. However, without those constraints, that value would certainly be higher as the algorithm would likely had increased the throughput rate of every machine to maximum.

Table 5.3: Production Without Algorithm

	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8
Production Without Algorithm	2520	2520	2520	1870	2395	2520	2435	2065
Balance	+120	+120	+120	-530	-5	+120	+35	-335

The last case worth-mentioning is week 4,from 22-01-2016 to 29-01-2016, where machine 1 was maintained at the beginning of the week. This case is interesting since without the algorithm all the machines would fail in that week (Figure 5.2) and the target would not be fulfilled (Table 5.3). Therefore, the algorithm outweighed the lack of production by increasing the load of the most productive machine, machine 3, anticipating its maintenance action, and working at maximum load after. Besides, it avoided a long maintenance action of machine 2 in the same week. Even though in a slower pace, machine 2 was able to produce some parts in that week and helped to reach the target.

To understand how systems performed, Tables 5.2 and 5.3 show the balance between parts produced and the weekly targets throughout the entire test duration.

While producing at normal rate, the systems produce per week 1520 parts, 120 more than the target which is set at 1400 parts. In the fist three weeks no failures occur in both tests, so, both produced 1520 parts. The first failure occurred in the fourth week, where the balance for the reference test, was -530 less than target and with algorithm + 30 than target.

Table 5.4: System Availability

	Availability
Without Algorithm	94.2%
With Algorithm	94.0%

As shown in Figure 5.2 in that week for the test without algorithm, the three machines went down for maintenance, resulting in a great production deficit. Whereas in the test with algorithm as previously explained, the algorithm manage to avoid a third maintenance, and produced slightly more than the target.

The following week, week 5, due to an imbalance in the implemented penalty's the optimized system excessively produced more 965 parts than the target. The reference test balance was of -5 parts less than the target.

This trend continues for the remaining weeks. Where the system with algorithm always comply with the target production, while the reference test, struggles to meet the requirements.

To proper evaluate each test, Overall Equipment Efficiency (OEE) was calculated. As previously explained in 2, the OEE is the ratio of fully productive time to planned production time.

To calculate the OEE, first is needed to determine its factors: Availability, Performance and Quality.

As in this document is assumed that the system only produce good parts, Quality factor is equal to 100 % for both tests.

The total system availability is given by the ratio between the run time of the system to its planned production time. As in Table 5.4, this factor gives a slight advantage to the test without algorithm. Since the last maintenance in the test was not completed before the end of simulation (Figure 5.2).

Table 5.5: System Performance

	Performance
Without Algorithm	93.5%
With Algorithm	106%

Performance factor compares the system cycle time with an ideal cycle time, being cycle time the average time to produce one part. In this case the ideal cycle time is defined as the cycle time of normal state load setting.

The results are shown in Table 5.5, the performance of the test with algorithm was 106%. This is due to the fact that the ideal cycle is set as the normal state cycle time. Which is lower than the algorithm solution cycle time. The reference test has a performance of 93.5%

With all the factors determined, it is now possible to calculate the OEE. Which, takes in account all system losses. As shown in Table 5.6, the test with algorithm has an OEE

Table 5.6: System Overall Equipment Effectiveness

	OEE
Without Algorithm	88.1%
With Algorithm	99.6%

of 99.6%. However, the test without algorithm performs 11.5% worse, with OEE equal to 88.1%.

Finally computaion-wise, both tests were run in a laptop with the following specification: Intel core i7-5500UU CPU @ 2.40GHz and 8.00GB of Random-access memory. For the reference test it took 78 seconds (1 minute and 18 seconds) and the test with algorithm took 84240 seconds (23 Hours and 24 minutes). This huge difference is mostly due to the optimization model where, the fitness function for each individual evaluated it has to run the simulation for the whole week, which drastically increases the computational cost.

In conclusion it is possible to prove that a joint approach of predictive maintenance models and throughput optimization models is beneficial for the performance of manufacturing systems.

Comparing each test availability (Table 5.4) and the total production of both systems (Tables 5.2 and 5.3) it is possible to conclude that with the same number of maintenance actions, the system with the algorithm was able to produce more parts than the reference test. Allowing the system with algorithm to have met always with the production target. Whereas, in the reference test, three out of eight weeks the system was not able to achieve the production target, which, could lead to major losses for the overall company profitability.

Chapter 6

Conclusions

6.1 Conclusions

As mentioned throughout this dissertation, the main objective was to create a joint approach consisting of a predictive maintenance and throughput optimization models, to reduce the maintenance related costs in a manufacturing system.

In order to solve the proposed problem, a Gradient Boosting Classifier was used to, based on continuous monitoring, predict incoming failures of assets in a manufacturing system. As failures were predicted, maintenance actions were scheduled and performed. However, as machine the whole system production rate was reduced. Therefore, a Genetic Algorithm was implemented to optimize the throughput rate of each machine, in order to, optimize the balance between production and degradation rate.

The proposed model was evaluated, by comparing the system with throughput optimization model and one reference system without it. The results show that the proposed model performed 11.5% better than the reference test according to the OEE metric. Overall system availability was similar, even though, the total production of the system with algorithm was higher than the reference test.

As previously stated, python was used as the main programming language. Despite the results obtained, it should be noted that with no future improvements this implementation is not applicable in a practical scenario. Due to high computational cost, this implementation is not scalable for many machine. Which makes it not feasible to implement in a real-life manufacturing system.

As final note, even though, the computational constraints, this document proved that there are benefits in integrating throughput optimization models in a manufacturing system. Which could mitigate the effect of maintenance action in the system total throughput. Which would help company's to reach their production targets, even though unscheduled maintenance actions occurred during production. Decreasing costumer satisfaction and decreasing maintenance related costs

6.2 Future Work

The main goal of this dissertation, to explore the benefits of integrating throughput optimization models in manufacturing systems, was successfully achieved. Nevertheless in such a broad topic there is always possible room to improve upon the proposed implementation and there always paths left to explore. Therefore, in a future work the following subjects could be further explored:

- The first, and the most obvious, is to optimize the proposed implementation, in order, to reduce it computational cost. And further test in a real-life system. Other optimization methods, such as Simulated Annealing and Tabu search, could be further explored and compared with the Genetic Algorithm.

- This implementation predicts a failure only one day prior it occurs. To allow manufactures more flexibility in maintenance and throughput optimizations. It could be interesting to increase this window, even though, successfully prediction rates could be lower.

- Lastly, in this document maintenance actions, reset each machine to a "as good as new " state. It would be interesting to explore other approaches that could allow imperfect maintenance to be performed. One suggestion is to explore a similar degradation model approach as [62] and integrate the throughput optimization model

Appendix A

A.1 Labelled Features

The following Table depicts the first records of the whole labelled features array analysed by the classifier.

Table A.1: Labeled Features

	machineID	datetime	voltmean__3h	rotatemean__3h	pressuremean__3h	vibrationmean__3h	voltsd__3h	rotatesd__3h
0	1	2015-01-02 06:00:00	180,133784	440,6083201	94,1379695	41,55154363	21,32273479	48,77051197
1	1	2015-01-02 09:00:00	176,3642932	439,349655	101,5532089	36,10558003	18,95221004	51,32963577
2	1	2015-01-02 12:00:00	160,3845679	424,3853157	99,59872201	36,09463702	13,04707951	13,70249554
3	1	2015-01-02 15:00:00	170,4724608	442,9339969	102,3805856	40,48300155	16,64235413	56,29044728
4	1	2015-01-02 18:00:00	163,2638057	468,9375583	102,7266479	40,92180163	17,42468821	38,6803798

	pressuresd__3h	vibrationsd__3h	voltmean__24h	rotatemean__24h	pressuremean__24h	vibrationmean__24h	voltsd__24h	rotatesd__24h
0	2,135683733	10,03720806	169,7338089	445,1798646	96,79711296	40,38515999	11,23312028	48,71739478
1	13,78927949	6,737739191	170,6148619	446,3648591	96,84978485	39,73682577	12,51940225	48,38507588
2	9,988608665	1,639962225	169,893965	447,0094075	97,71559982	39,49837385	13,3703566	42,43231716
3	3,305738927	8,854145348	171,2434437	444,2335635	96,66606039	40,22936989	13,2992806	41,34612144
4	9,10577492	3,060780703	170,792486	448,4404372	95,76683804	40,05521356	13,95451751	43,49023354

	pressuresd__24h	vibrationsd__24h	error1count	error2count	error3count	error4count	error5count	comp1	comp2	comp3
0	10,07988023	5,853208563	0	0	0	0	0	20	215	155
1	10,17153979	6,163230823	0	0	0	0	0	20,125	215,125	155,125
2	9,471669478	6,195075681	0	0	0	0	0	20,25	215,25	155,25
3	8,73122875	5,687944234	0	0	0	0	0	20,375	215,375	155,375
4	8,06165254	5,89806871	0	0	0	0	0	20,5	215,5	155,5

	comp4	model	age	failure
0	170	model3	18	none
1	170,125	model3	18	none
2	170,25	model3	18	none
3	170,375	model3	18	none
4	170,5	model3	18	none

References

[1] Maurizio Bevilacqua and Marcello Braglia. The analytic hierarchy process applied to maintenance strategy selection. *Reliability Engineering & System Safety*, 70(1):71–83, 2000.

[2] Damjan Maletič, Matjaž Maletič, Basim Al-Najjar, and Boštjan Gomišček. The role of maintenance in improving company's competitiveness and profitability: a case study in a textile company. *Journal of Manufacturing Technology Management*, 25(4):441–456, 2014.

[3] Ming-Yi You, Fang Liu, Wen Wang, and Guang Meng. Statistically planned and individually improved predictive maintenance management for continuously monitored degrading systems. *IEEE Transactions on Reliability*, 59(4):744–753, 2010.

[4] Nagi Gebraeel. Sensory-updated residual life distributions for components with exponential degradation patterns. *IEEE Transactions on Automation Science and Engineering*, 3(4):382–393, 2006.

[5] Zimin Yang, Dragan Djurdjanovic, and Jun Ni. Maintenance scheduling for a manufacturing system of machines with adjustable throughput. *IIE transactions*, 39(12):1111–1125, 2007.

[6] Zimin Max Yang, Dragan Djurdjanovic, and Jun Ni. Maintenance scheduling in manufacturing systems based on predicted machine degradation. *Journal of intelligent manufacturing*, 19(1):87–98, 2008.

[7] K Efthymiou, A Pagoropoulos, N Papakostas, D Mourtzis, and G Chryssolouris. Manufacturing systems complexity review: challenges and outlook. *Procedia CIRP*, 3:644–649, 2012.

[8] Zhuming M Bi, Sherman YT Lang, Weiming Shen, and Lihui Wang. Reconfigurable manufacturing systems: the state of the art. *International Journal of Production Research*, 46(4):967–992, 2008.

[9] Yoram Koren. *The global manufacturing revolution: product-process-business integration and reconfigurable systems*, volume 80. John Wiley & Sons, 2010.

[10] S Jack Hu. Evolving paradigms of manufacturing: from mass production to mass customization and personalization. *Procedia CIRP*, 7:3–8, 2013.

[11] Carmen Magar Anshuk Gandhi and Roger Roberts. How technology can drive the next wave of mass customization. 2009.

[12] José Soares. Models for innovative factory layout design techniques and adaptive reconfiguration in the automotive industry, 2016.

[13] Mostafa G Mehrabi, A Galip Ulsoy, Yoram Koren, and Peter Heytler. Trends and perspectives in flexible and reconfigurable manufacturing systems. *Journal of Intelligent manufacturing*, 13(2):135–146, 2002.

[14] Yoram Koren and Moshe Shpitalni. Design of reconfigurable manufacturing systems. *Journal of manufacturing systems*, 29(4):130–141, 2010.

[15] Kathryn E Stecke. Design, planning, scheduling, and control problems of flexible manufacturing systems. *Annals of Operations research*, 3(1):1–12, 1985.

[16] Andrew Kusiak. *Computational intelligence in design and manufacturing*. John Wiley & Sons, 2000.

[17] E Oztemel. Intelligent manufacturing systems. *Artificial intelligence techniques for networked manufacturing enterprises management*, pages 1–41, 2010.

[18] Weiming Shen, Qi Hao, Hyun Joong Yoon, and Douglas H Norrie. Applications of agent-based systems in intelligent manufacturing: An updated review. *Advanced engineering INFORMATICS*, 20(4):415–431, 2006.

[19] Joseph H Saleh and Ken Marais. Highlights from the early (and pre-) history of reliability engineering. *Reliability engineering & system safety*, 91(2):249–256, 2006.

[20] Patrick DT O'connor, Patrick O'Connor, and Andre Kleyner. *Practical reliability engineering*. John Wiley & Sons, 2012.

[21] Mohammad Modarres, Mark P Kaminskiy, and Vasiliy Krivtsov. *Reliability engineering and risk analysis: a practical guide*. CRC press, 2016.

[22] Õrjan Ljungberg. Measurement of overall equipment effectiveness as a basis for tpm activities. *International Journal of Operations & Production Management*, 18(5):495–507, 1998.

[23] Susana Aguiar, Rui Pinto, Joao Reis, and Gil Manuel Gonçalves. Life-cycle approach to extend equipment re-use in flexible manufacturing. In *INTELLI 2016, The Fifth International Conference on Intelligent Systems and Applications*, 2016.

[24] Bulent Dal, Phil Tugwell, and Richard Greatbanks. Overall equipment effectiveness as a measure of operational improvement–a practical analysis. *International Journal of Operations & Production Management*, 20(12):1488–1502, 2000.

[25] S Nakaiima. Introduction to total productive maintenance (tpm), 1988.

[26] ReliaSoft Corporation. Mttf, mtbf, mean time between replacements and mtbf with scheduled replacements, May 2017. URL: http://www.weibull.com/hotwire/issue94/relbasics94.htm [last accessed 2017-05-12].

[27] Charles E Ebeling. *An introduction to reliability and maintainability engineering*. Tata McGraw-Hill Education, 2004.

[28] Magne Vollan Aarset. How to identify a bathtub hazard rate. *IEEE Transactions on Reliability*, 36(1):106–108, 1987.

[29] Xingguo Xiong, Yu-Liang Wu, and Wen-Ben Jone. Reliability model for mems accelerometers. In *Novel Algorithms and Techniques In Telecommunications, Automation and Industrial Electronics*, pages 261–266. Springer, 2008.

[30] Dennis J. Wilkins. The bathtub curve and product failure behavior part one - the bathtub curve, infant mortality and burn-in, May 2017. URL: http://www.weibull.com/hotwire/issue94/relbasics94.htm [last accessed 2017-05-12].

[31] Andrew N O'Connor. *Probability distributions used in reliability engineering*. RiAC, 2011.

[32] ReliaSoft Corporation. The exponential distribution, May 2017. URL: http://reliawiki.org/index.php/The_Exponential_Distribution#The_Effect_of_lambda_and_gamma_on_the_Failure_Rate_Function [last accessed 2017-05-12].

[33] Kailash C Kapur and Michael Pecht. *Reliability engineering*. John Wiley & Sons, 2014.

[34] Melih Fidanoglu, Ufuk Ungor, Ibrahim Ozkol, and Guven Komurgoz. Application of weibull distribution method for aircraft component life estimation in civil aviation sector. *Journal of Traffic and Logistics Engineering Vol*, 5(1), 2017.

[35] Jaouher Ben Ali, Brigitte Chebel-Morello, Lotfi Saidi, Simon Malinowski, and Farhat Fnaiech. Accurate bearing remaining useful life prediction based on weibull distribution and artificial neural network. *Mechanical Systems and Signal Processing*, 56:150–172, 2015.

[36] Abdellatif Bey-Temsamani, Marc Engels, Andy Motten, Steve Vandenplas, and Agusmian P Ompusunggu. A practical approach to combine data mining and prognostics for improved predictive maintenance. *Data Min. Case Stud*, 36, 2009.

[37] Zhigang Tian. An artificial neural network method for remaining useful life prediction of equipment subject to condition monitoring. *Journal of Intelligent Manufacturing*, 23(2):227–237, 2012.

[38] MI Mazhar, S Kara, and H Kaebernick. Remaining life estimation of used components in consumer products: Life cycle data analysis by weibull and artificial neural networks. *Journal of operations management*, 25(6):1184–1193, 2007.

[39] George J Vachtsevanos, Frank Lewis, Andrew Hess, and Biqing Wu. *Intelligent fault diagnosis and prognosis for engineering systems*. Wiley Online Library, 2006.

[40] Nima Gorjian, Lin Ma, Murthy Mittinty, Prasad Yarlagadda, and Yong Sun. A review on degradation models in reliability analysis. In *Engineering Asset Lifecycle Management*, pages 369–384. Springer, 2010.

[41] JZ Sikorska, Melinda Hodkiewicz, and Lin Ma. Prognostic modelling options for remaining useful life estimation by industry. *Mechanical Systems and Signal Processing*, 25(5):1803–1836, 2011.

[42] Ying Peng, Ming Dong, and Ming Jian Zuo. Current status of machine prognostics in condition-based maintenance: a review. *The International Journal of Advanced Manufacturing Technology*, 50(1):297–313, 2010.

[43] inc The Mathworks. Fuzzy inference process. URL: https://www.mathworks.com/help/fuzzy/fuzzy-inference-process.html?requestedDomain=www.mathworks.com [last accessed 2017-05-12].

[44] Wilson Q Wang, M Farid Golnaraghi, and Fathy Ismail. Prognosis of machine health condition using neuro-fuzzy systems. *Mechanical Systems and Signal Processing*, 18(4):813–831, 2004.

[45] B Satish and NDR Sarma. A fuzzy bp approach for diagnosis and prognosis of bearing faults in induction motors. In *Power Engineering Society General Meeting, 2005. IEEE*, pages 2291–2294. IEEE, 2005.

[46] Jianhui Luo, Madhavi Namburu, Krishna Pattipati, Liu Qiao, Masayuki Kawamoto, and SACS Chigusa. Model-based prognostic techniques [maintenance applications]. In *AUTOTESTCON 2003. IEEE Systems Readiness Technology Conference. Proceedings*, pages 330–340. IEEE, 2003.

[47] Khaoula Tidriri, Nizar Chatti, Sylvain Verron, and Teodor Tiplica. Bridging data-driven and model-based approaches for process fault diagnosis and health monitoring: A review of researches and future challenges. *Annual Reviews in Control*, 42:63–81, 2016.

[48] M Krupa. Methods of technical prognosis-review. *Brno University of Technology*, 2010.

[49] Rodney K Singleton, Elias G Strangas, and Selin Aviyente. Extended kalman filtering for remaining-useful-life estimation of bearings. *IEEE Transactions on Industrial Electronics*, 62(3):1781–1790, 2015.

[50] Reuben Lim and David Mba. Condition monitoring and remaining useful life prediction using switching kalman filters. *International Journal of Strategic Engineering Asset Management 8*, 2(1):22–36, 2014.

[51] Man Shan Kan, Andy CC Tan, and Joseph Mathew. A review on prognostic techniques for non-stationary and non-linear rotating systems. *Mechanical Systems and Signal Processing*, 62:1–20, 2015.

[52] Jingliang Zhang and Jay Lee. A review on prognostics and health monitoring of li-ion battery. *Journal of Power Sources*, 196(15):6007–6014, 2011.

[53] Zhi-Jie Zhou and Chang-Hua Hu. An effective hybrid approach based on grey and arma for forecasting gyro drift. *Chaos, Solitons & Fractals*, 35(3):525–529, 2008.

[54] Wei Wu, Jingtao Hu, and Jilong Zhang. Prognostics of machine health condition using an improved arima-based prediction method. In *Industrial Electronics and Applications, 2007. ICIEA 2007. 2nd IEEE Conference on*, pages 1062–1067. Ieee, 2007.

[55] Judea Pearl. Bayesian networks. *Department of Statistics, UCLA*, 2011.

[56] B Jones, Ian Jenkinson, Z Yang, and Jin Wang. The use of bayesian network modelling for maintenance planning in a manufacturing industry. *Reliability Engineering & System Safety*, 95(3):267–277, 2010.

[57] Martin Neil and David Marquez. Availability modelling of repairable systems using bayesian networks. *Engineering Applications of Artificial Intelligence*, 25(4):698–704, 2012.

[58] Zhi-Sheng Ye and Min Xie. Stochastic modelling and analysis of degradation for highly reliable products. *Applied Stochastic Models in Business and Industry*, 31(1):16–32, 2015.

[59] Jihong Yan. *Machinery Prognostics and Prognosis Oriented Maintenance Management*. John Wiley & Sons, 2014.

[60] Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian network classifiers. *Machine learning*, 29(2-3):131–163, 1997.

[61] Joris Hulst. Modeling physiological processes with dynamic bayesian networks, 2006.

[62] Anselm Lorenzoni and Michael Kempf. Degradation processes modelled with dynamic bayesian networks. In *Industrial Informatics (INDIN), 2015 IEEE 13th International Conference on*, pages 1694–1699. IEEE, 2015.

[63] Jinqiu Hu, Laibin Zhang, Lin Ma, and Wei Liang. An integrated safety prognosis model for complex system based on dynamic bayesian network and ant colony algorithm. *Expert Systems with Applications*, 38(3):1431–1446, 2011.

[64] Ken R McNaught and Adam Zagorecki. Using dynamic bayesian networks for prognostic modelling to inform maintenance decision making. In *Industrial Engineering and Engineering Management, 2009. IEEM 2009. IEEE International Conference on*, pages 1155–1159. IEEE, 2009.

[65] Susana Ferreiro, Aitor Arnaiz, Basilio Sierra, and Itziar Irigoien. Application of bayesian networks in prognostics for a new integrated vehicle health management concept. *Expert Systems with Applications*, 39(7):6402–6418, 2012.

[66] Pedro A Pérez Ramírez and Ingrid Bouwer Utne. Use of dynamic bayesian networks for life extension assessment of ageing systems. *Reliability Engineering & System Safety*, 133:119–136, 2015.

[67] Jianzhong Sun, Hongfu Zuo, Wenbin Wang, and Michael G Pecht. Prognostics uncertainty reduction by fusing on-line monitoring data based on a state-space-based degradation model. *Mechanical Systems and Signal Processing*, 45(2):396–407, 2014.

[68] Ming Dong and David He. A segmental hidden semi-markov model (hsmm)-based diagnostics and prognostics framework and methodology. *Mechanical systems and signal processing*, 21(5):2248–2266, 2007.

[69] Xiao-Sheng Si, Wenbin Wang, Chang-Hua Hu, and Dong-Hua Zhou. Remaining useful life estimation–a review on the statistical data driven approaches. *European journal of operational research*, 213(1):1–14, 2011.

[70] Khanh Le Son, Mitra Fouladirad, Anne Barros, Eric Levrat, and Benoît Iung. Remaining useful life estimation based on stochastic deterioration models: A comparative study. *Reliability Engineering & System Safety*, 112:165–175, 2013.

[71] Shun-Zheng Yu. Hidden semi-markov models. *Artificial intelligence*, 174(2):215–243, 2010.

[72] Qinming Liu, Ming Dong, and Ying Peng. A novel method for online health prognosis of equipment based on hidden semi-markov model using sequential monte carlo methods. *Mechanical Systems and Signal Processing*, 32:331–348, 2012.

[73] Ying Peng and Ming Dong. A prognosis method using age-dependent hidden semi-markov model for equipment health prediction. *Mechanical Systems and Signal Processing*, 25(1):237–252, 2011.

[74] Ying Peng and Ming Dong. A hybrid approach of hmm and grey model for age-dependent health prediction of engineering assets. *Expert Systems with Applications*, 38(10):12946–12953, 2011.

[75] Robert Gao, Lihui Wang, Roberto Teti, David Dornfeld, Soundar Kumara, Masahiko Mori, and M Helu. Cloud-enabled prognosis for manufacturing. *CIRP Annals-Manufacturing Technology*, 64(2):749–772, 2015.

[76] Farzaneh Ahmadzadeh and Jan Lundberg. Remaining useful life estimation. *International Journal of System Assurance Engineering and Management*, 5(4):461–474, 2014.

[77] João Correia. Nonlinear analog equalizer for 5th generation communication systems, 2017.

[78] Anastasios P Vassilopoulos and Raman Bedi. Adaptive neuro-fuzzy inference system in modelling fatigue life of multidirectional composite laminates. *Computational Materials Science*, 43(4):1086–1093, 2008.

[79] Zhigang Tian, Tongdan Jin, Bairong Wu, and Fangfang Ding. Condition based maintenance optimization for wind power generation systems under continuous monitoring. *Renewable Energy*, 36(5):1502–1509, 2011.

[80] Abd Kadir Mahamad, Sharifah Saon, and Takashi Hiyama. Predicting remaining useful life of rotating machinery based artificial neural network. *Computers & Mathematics with Applications*, 60(4):1078–1087, 2010.

[81] Arnaz Malhi, Ruqiang Yan, and Robert X Gao. Prognosis of defect propagation based on recurrent neural networks. *IEEE Transactions on Instrumentation and Measurement*, 60(3):703–711, 2011.

[82] Sze-jung Wu, Nagi Gebraeel, Mark A Lawley, and Yuehwern Yih. A neural network integrated decision support system for condition-based optimal predictive maintenance policy. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 37(2):226–236, 2007.

[83] Farzaneh Ahmadzadeh and Jan Lundberg. Remaining useful life prediction of grinding mill liners using an artificial neural network. *Minerals Engineering*, 53:1–8, 2013.

[84] Damjan Maletič, Matjaž Maletič, Basim Al-Najjar, and Boštjan Gomišček. The role of maintenance in improving company's competitiveness and profitability: a case study in a textile company. *Journal of Manufacturing Technology Management*, 25(4):441–456, 2014.

[85] Albert HC Tsang. Condition-based maintenance: tools and decision making. *Journal of Quality in Maintenance Engineering*, 1(3):3–17, 1995.

[86] Fatih Camci. System maintenance scheduling with prognostics information using genetic algorithm. *IEEE Transactions on reliability*, 58(3):539–552, 2009.

[87] Fiix. Preventive maintenance (pm). URL: https://www.fiixsoftware.com/maintenance-strategies/preventative-maintenance/ [last accessed 2017-11-12].

[88] Rosmaini Ahmad and Shahrul Kamaruddin. An overview of time-based and condition-based maintenance in industrial application. *Computers & Industrial Engineering*, 63(1):135–149, 2012.

[89] Andrew KS Jardine and Albert HC Tsang. *Maintenance, replacement, and reliability: theory and applications*. CRC press, 2013.

[90] Xiaojun Zhou, Lifeng Xi, and Jay Lee. Reliability-centered predictive maintenance scheduling for a continuously monitored system subject to degradation. *Reliability Engineering & System Safety*, 92(4):530–534, 2007.

[91] Mohammad Doostparast, Farhad Kolahan, and Mahdi Doostparast. A reliability-based approach to optimize preventive maintenance scheduling for coherent systems. *Reliability Engineering & System Safety*, 126:98–106, 2014.

[92] Hoang Pham and Hongzhou Wang. Imperfect maintenance. *European journal of operational research*, 94(3):425–438, 1996.

[93] Alexandre Muller, Marie-Christine Suhner, and Benoît Iung. Formalisation of a new prognosis model for supporting proactive maintenance implementation on industrial system. *Reliability Engineering & System Safety*, 93(2):234–253, 2008.

[94] Phuc Do, Alexandre Voisin, Eric Levrat, and Benoit Iung. A proactive condition-based maintenance strategy with both perfect and imperfect maintenance actions. *Reliability Engineering & System Safety*, 133:22–32, 2015.

[95] Mazhar Ali Khan Malik. Reliable preventive maintenance scheduling. *AIIE transactions*, 11(3):221–228, 1979.

[96] Mayank Pandey, Ming J Zuo, Ramin Moghaddass, and MK Tiwari. Selective maintenance for binary systems under imperfect repair. *Reliability Engineering & System Safety*, 113:42–51, 2013.

[97] Hongzhou Wang. A survey of maintenance policies of deteriorating systems. *European journal of operational research*, 139(3):469–489, 2002.

[98] Javid Koochaki, Jos AC Bokhorst, Hans Wortmann, and Warse Klingenberg. Condition based maintenance in the context of opportunistic maintenance. *International Journal of Production Research*, 50(23):6918–6929, 2012.

[99] T Nakagawa and DNP Murthy. Optimal replacement policies for a two-unit system with failure interactions. *RAIRO-Operations Research*, 27(4):427–438, 1993.

[100] JS Dagpunar. A maintenance model with opportunities and interrupt replacement options. *Journal of the Operational Research Society*, pages 1406–1409, 1996.

[101] Fangfang Ding and Zhigang Tian. Opportunistic maintenance optimization for wind turbine systems considering imperfect maintenance actions. *International Journal of Reliability, Quality and Safety Engineering*, 18(05):463–481, 2011.

[102] Zia A Yamayee. Maintenance scheduling: description, literature survey, and interface with overall operations scheduling. *IEEE Transactions on Power Apparatus and Systems*, (8):2770–2779, 1982.

[103] Aurélien Froger, Michel Gendreau, Jorge E Mendoza, Éric Pinson, and Louis-Martin Rousseau. Maintenance scheduling in the electricity industry: A literature review. *European Journal of Operational Research*, 251(3):695–706, 2016.

[104] Edmund K Burke, JA Clarke, and Alistair J Smith. Four methods for maintenance scheduling. In *Artificial Neural Nets and Genetic Algorithms*, pages 264–269. Springer, 1998.

[105] Anil Sharma, GS Yadava, and SG Deshmukh. A literature review and future perspectives on maintenance optimization. *Journal of Quality in Maintenance Engineering*, 17(1):5–25, 2011.

[106] Jason Brownlee. *Clever algorithms: nature-inspired programming recipes*. Jason Brownlee, 2011.

[107] José L Ribeiro Filho, Philip C Treleaven, and Cesare Alippi. Genetic-algorithm programming environments. *Computer*, 27(6):28–43, 1994.

[108] Marzio Marseguerra, Enrico Zio, and Luca Podofillini. Condition-based maintenance optimization by means of genetic algorithms and monte carlo simulation. *Reliability Engineering & System Safety*, 77(2):151–165, 2002.

[109] Manoj Kumar, Mohammad Husian, Naveen Upreti, and Deepti Gupta. Genetic algorithm: Review and application. 2, 01 2010.

[110] Sanghamitra Bandyopadhyay and Sriparna Saha. Some single-and multiobjective optimization techniques. In *Unsupervised Classification*, pages 17–58. Springer, 2013.

[111] Brad L Miller, David E Goldberg, et al. Genetic algorithms, tournament selection, and the effects of noise. *Complex systems*, 9(3):193–212, 1995.

[112] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.

[113] E Moradi, SMT Fatemi Ghomi, and Mostafa Zandieh. Bi-objective optimization research on integrated fixed time interval preventive maintenance and production for scheduling flexible job-shop problem. *Expert systems with applications*, 38(6):7169–7178, 2011.

[114] F Pezzella, G Morganti, and G Ciaschetti. A genetic algorithm for the flexible job-shop scheduling problem. *Computers & Operations Research*, 35(10):3202–3212, 2008.

[115] Sai Ho Chung, Henry CW Lau, George TS Ho, and Wai Hung Ip. Optimization of system reliability in multi-factory production networks by maintenance approach. *Expert Systems with Applications*, 36(6):10188–10196, 2009.

[116] Christophe Varnier and Noureddine Zerhouni. Scheduling predictive maintenance in flow-shop. In *Prognostics and System Health Management (PHM), 2012 IEEE Conference on*, pages 1–6. IEEE, 2012.

[117] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[118] Stéfan van der Walt, S Chris Colbert, and Gael Varoquaux. The numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, 13(2):22–30, 2011.

[119] Wes McKinney. pandas: a foundational python library for data analysis and statistics.

[120] Félix-Antoine Fortin, François-Michel De Rainville, Marc-André Gardner, Marc Parizeau, and Christian Gagné. DEAP: Evolutionary algorithms made easy. *Journal of Machine Learning Research*, 13:2171–2175, jul 2012.

[121] K. Goebel A. Saxena. Turbofan engine degradation simulation data set. *NASA Ames Prognostics Data Repository*, 2008.

[122] Mary Wahl. Predictive maintenance modelling guide python notebook, 2016. URL: `https://gallery.cortanaintelligence.com/Notebook/Predictive-Maintenance-Modelling-Guide-Python-Notebook-1` [last accessed 2018-01-03].

[123] Andrew Ng. Machine learning and ai via brain simulations, 2013.

[124] Gustavo HT Ribeiro, Paulo SG de M Neto, George DC Cavalcanti, and Ren Tsang. Lag selection for time series forecasting using particle swarm optimization. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pages 2437–2444. IEEE, 2011.

[125] Andreas Mayr, Harald Binder, Olaf Gefeller, Matthias Schmid, et al. The evolution of boosting algorithms. *Methods of information in medicine*, 53(6):419–427, 2014.

[126] Robert E Schapire. The boosting approach to machine learning: An overview. In *Nonlinear estimation and classification*, pages 149–171. Springer, 2003.

[127] Alexey Natekin and Alois Knoll. Gradient boosting machines, a tutorial. *Frontiers in neurorobotics*, 7, 2013.

[128] Jerome H Friedman. Stochastic gradient boosting. *Computational Statistics & Data Analysis*, 38(4):367–378, 2002.

[129] Fatih Camci. System maintenance scheduling with prognostics information using genetic algorithm. *IEEE Transactions on reliability*, 58(3):539–552, 2009.

[130] Olympia Roeva, Stefka Fidanova, and Marcin Paprzycki. Influence of the population size on the genetic algorithm performance in case of cultivation process modelling. In *Computer Science and Information Systems (FedCSIS), 2013 Federated Conference on*, pages 371–376. IEEE, 2013.

[131] Anupriya Shukla, Hari Mohan Pandey, and Deepti Mehrotra. Comparative review of selection techniques in genetic algorithm. In *Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE), 2015 International Conference on*, pages 515–519. IEEE, 2015.

[132] Noraini Mohd Razali, John Geraghty, et al. Genetic algorithm performance with different selection strategies in solving tsp. In *Proceedings of the world congress on engineering*, volume 2, pages 1134–1139, 2011.