FH Vorarlberg
University of Applied Sciences

Predictive Maintenance for Test Tower

Analysis of Suitability of a Predictive Maintenance Strategy for an Electrical Test Tower

Master Thesis
Submitted in Fulfillment of the Degree

Master of Science in Engineering (MSc)

University of Applied Sciences Vorarlberg
Mechatronics

Submitted to
Steffen Finck

Handed in by
Gabdelgaziz Sayfutdinov
Dornbirn, August 31, 2021

Kurzreferat

Vorausschauende Wartung für elektrische Testtürme

Die Firma OMICRON electronics GmbH führt Gesamtgerätetests für ihre Endgeräte auf Testtürmen durch. Diese Tests bestehen aus mehreren Subtests, die Messergebnisse zurückliefern und diese in einer Datenbank abspeichern. Das Ziel dieser Arbeit ist es die Messergebnisse der Subtests zu analysieren und festzustellen, inwiefern eine Implementierung einer vorausschauenden Wartung für die Testtürme möglich ist. Mit solch einer vorausschauenden Wartung wäre es möglich die verbleibende Nutzungsdauer zu bestimmen und die Abnutzung des Turms zu bestimmen.
Die Annahme ist, dass die Hauptursache bzw. die Hauptquelle der Abnutzung des Turms die Relais sind. Mit dieser Annahme ist es möglich den Turm mithilfe einer Verlässlichkeitsanalyze modellgetrieben zu modellieren.
Die datengetriebene Modellierung des Testturms besteht aus mehreren Schritten. Zuerst wird der gesamte Datensatz gesäubert und komprimiert, indem redundante Datensätze entfernt werden. Zusätzlich wird der Datensatz optimiert, indem nur Subtests mit den besten Trendabilitäts- und Monotonitätswerten behalten werden und es wird ein Ranking der Subtests durchgefürt. Anschließend werden die verbleibenden Subtests auf die Trendabilität und Monotonität hin getestet, die zeigen, dass der gesamte Datensatz schlechtes Trendverhalten hat und somit eine vorausschauende Wartung nicht möglich ist.
Indem man das Ranking der Verlässlichkeitsanalyze und der Subtests des datengetriebenen Modells verglichen hat, konnte festgestellt werden, dass die Annahme der Relais als Hauptfehlerquelle falsch ist.
Eine mögliche Anomalie-Detektion für eine vorausschauende Wartung wurde getestet, was gezeigt hat, dass solch eine Detektion für die Testtürme nicht möglich ist.
Die Implementierbarkeit einer vorausschauenden Wartung für Testtürme und andere OMICRON Geräte wird diskutiert und zusätzliche Vorschläge für mögliche zukünftige Implementierung für vorausschauende Wartungen werden gemacht.

Abstract

Predictive maintenance for electrical testing tower

The company OMICRON electronics GmbH performs run-in tests for its devices on test towers. Those tests consist of a multitude of subtests which all return a measurement value. Those results are tracked and stored in a database. The goal is to analyze the data of the test towers subtests and evaluate the possibility of implementing a predictive maintenance system in order to be able to predict the RUL and quantify the degradation of the test tower.
By assuming that the main degradation source are the relays, a reliability modelling is performed which represents the modell-driven approach.
The data-driven modelling of the test tower consists of multiple steps. Firstly, the data is cleaned and compromised by removing redundances and optimizing for the best subtests where a subtest is rated as good if the trendability and monotonicity metric values are above specified thresholds. In a second step, the trend behaviours of the subtests are analyzed and ranked which illustrated that none of the subtests contained usable trend behaviour thus making an implementation of a PdM system impossible.
By using the ranking, the data-driven model was compared with the reliability model which showed that the assumption of the relays being the main error source was inaccurate.
An analysis of a possible anomaly detection model for a PdM is evaluated which showed that an anomaly detection is not possible for the test towers either.
The implementability of PdM for test towers and other OMICRON devices is discussed and followed up with proposals for future PdM implementations as well as additional analytical analyzes that can be performed for the test towers.

Contents

List of figures 8

List of tables 10

List of abbreviations 11

1 Introduction 12
- 1.1 What is predictive maintenance? . . . 12
- 1.2 Why to use predictive maintenance? . . . 13
  - 1.2.1 Benefits of PdM . . . 13
  - 1.2.2 Drawbacks of PdM . . . 14
- 1.3 When to use predictive maintenance? . . . 14
  - 1.3.1 Use it! . . . 14
  - 1.3.2 Don't use it! . . . 15
  - 1.3.3 Conclusion . . . 15
- 1.4 Requirements for PdM . . . 16
  - 1.4.1 Suitability of systems for PdM . . . 16
  - 1.4.2 Requirements for datasets . . . 17
- 1.5 Workflow of PdM . . . 17

2 Research question 20

3 System setup 21
- 3.1 Test tower . . . 21
- 3.2 DUTs . . . 24
- 3.3 Unit test . . . 25
- 3.4 Database . . . 26
- 3.5 Software . . . 26

4 Modelling of test tower 27
- 4.1 Limitations . . . 27
- 4.2 Distribution of relays in subtests . . . 27
- 4.3 Reliability analysis . . . 30
- 4.4 Simulation results for the test tower . . . 31
- 4.5 Simulation results for separate subtests . . . 33

5 Data preparation 34
- 5.1 Selected PdM model . . . 34

5.2 Requirements for final dataset . . . 34
5.3 General information . . . 34
5.4 Data presentation . . . 34
5.5 Data fetching . . . 38
5.6 Feature engineering . . . 40
5.6.1 Tower name feature . . . 40
5.6.2 Data cleaning . . . 40
5.6.3 Serial number feature . . . 41
5.6.4 Error feature . . . 41
5.6.5 Timestamp feature . . . 44
5.6.6 Moving average . . . 45
5.6.7 Time domain features . . . 48
5.6.8 Frequency domain features . . . 48
5.6.9 Time-frequency features . . . 48
5.7 Theoretical background - comparison techniques for features and signals 49
5.7.1 Dynamic time warping . . . 49
5.7.2 Monotonicity . . . 49
5.7.3 Trendability . . . 50
5.7.4 Thresholds for trend evaluations . . . 51
5.8 Dimensionality reduction and abstraction of the unit test . . . 54
5.8.1 Abstraction of the unit test . . . 54
5.8.2 Dimensionality reduction . . . 69
5.9 Final dataset . . . 69

6 Data exploration **71**
6.1 Exploration . . . 71
6.2 Trend analysis . . . 78
6.3 ETT1 vs. ETT2 . . . 79
6.4 Trend analysis vs. Tower Modelling . . . 81

7 Implementation of PdM **84**
7.1 Data preparation for ML . . . 84
7.2 RUL prediction . . . 84
7.3 Anomaly detection . . . 85

8 Proposed further action for PdM **88**

9 Conclusion **90**

Bibliography **92**

Appendix **97**
9.1 Source code . . . 98
9.1.1 Database access . . . 98

9.1.2 Trendability analysis 100
9.1.3 Monotonicity analysis 100

Statement of Affirmation **102**

List of Figures

1.1 Maintenance strategies (own illustration, based on [32][42]) 13
1.2 Common predictive technology applications [15] 16
1.3 PdM workflow 17
1.4 Lifetime data 18
1.5 Run-to-failure data 19
1.6 Threshold data 19

3.1 Test tower setup 21
3.2 Actual test tower setup 22
3.3 Modules of the test tower 23
3.4 CMC430, an ultra-portable Protection Test Set and Calibrator [26][10] 24
3.5 ARCO400, a universal test set for recloser controls[27] 24
3.6 CMC430 unit test structure - level 1 25

4.1 Used relays for subtests in CMC430 29
4.2 Distribution of relay types 30
4.3 Reliability for electrical endurance 32
4.4 Reliability for mechanical endurance 32
4.5 Comparison of mechanical and electrical endurance 33

5.1 Subtest data from DB workflow 38
5.2 Known tower module IDs and its corresponding test tower 40
5.3 Filled tower names 41
5.4 Data after removing production series dependency 42
5.5 Creation of 'relative error re first use'-feature 43
5.6 Result of 'relative error re first use'-feature 44
5.7 Filtering effect after summing up into daily samples 45
5.8 Distribution plot for filtering effect after summing up into daily samples 46
5.9 Filtering effect after SMA smoothing 47
5.10 Distribution plot for filtering effect after SMA smoothing 47
5.11 Euclidean Matching vs. DTW Matching[47] 50
5.12 Trendability and monotonicity results for motor degradation 52
5.13 Plots for the `s4`, `s7` and `s8` parameters of the motor 52
5.14 Trendability and monotonicity results for wind turbine degradation . . 53
5.15 Plots for the `Mean`, `Skew` and `Kurtosis` parameters of the wind turbine 53
5.16 DTW matching for M2Sub2 parameter B 55
5.17 Trendability for the different channel classes 57

5.18 M1 Sub1 signals . . . 58
5.19 M1 Sub1 trendability and monotonicity analysis . . . 59
5.20 M2 Sub1 trendability and monotonicity analysis . . . 59
5.21 M2 Sub1 signals . . . 60
5.22 M2 Sub2 signals . . . 61
5.23 M2 Sub2 trendability and monotonicity analysis . . . 61
5.24 M2 Sub3 signals . . . 62
5.25 M2 Sub3 trendability and monotonicity analysis . . . 62
5.26 M2 Sub4 signals . . . 63
5.27 M2 Sub4 trendability and monotonicity analysis . . . 63
5.28 M2 Sub5 signals . . . 64
5.29 M2 Sub5 trendability and monotonicity analysis . . . 64
5.30 M2 Sub6 signals . . . 65
5.31 M2 Sub6 trendability and monotonicity analysis . . . 65
5.32 M3 Sub1 . . . 67
5.33 M3 Sub2 . . . 67
5.34 M4 Sub1 . . . 68
5.35 M4 Sub2 . . . 68

6.1 Relative error for the selected subtests . . . 72
6.2 Standard deviation of relative error for the selected subtests . . . 72
6.3 Frequency domain analysis with Fourier transform for the selected subtests 73
6.4 Relative error re first use for the selected subtests . . . 74
6.5 Standard deviation for the selected subtests . . . 75
6.6 Skewness for the selected subtests . . . 75
6.7 Kurtosis for the selected subtests . . . 75
6.8 Testing of DUT happens in chunks . . . 76
6.9 Relative error re first use including the time points at which the subtest was unused for 7 days . . . 77
6.10 Relative error re first use including the time points at which the subtest was unused for 21 days . . . 77
6.11 Comparison of subtests in monotonicity and trendability for ETT1 . . 78
6.12 Comparison of subtests in monotonicity and trendability for ETT2 . . 79
6.13 Comparison of ETT1 and ETT2 in their trendability . . . 80
6.14 Comparison of ETT1 and ETT2 in their monotonicity . . . 80
6.15 Linear regression of the M2Sub3 subtest . . . 82

7.1 Anomaly detection with IQR . . . 85
7.2 Anomaly detection with K-Means . . . 87

List of Tables

3.1 Test tower components 23
3.2 Relevant CMC430 modules in the context of this research 24

4.1 Relay types and their corresponding indices 28
4.2 Used relay types in test tower 28

5.1 Summary of subtests for CMC430 35
5.2 Aliases for modules and subtests 36
5.3 Aliases for parameters 37
5.4 Time domain features 48
5.5 Overview of time distances D for each subtest with multiple channels . 56
5.6 Overview of subtests after removing channel dependency 57
5.7 Overview of subtests after removing parameter dependency and result types 66
5.8 Overview of the final subtests to be used for PdM 70

6.1 Dickey-Fuller test result 73
6.2 Number of relays for each subtest 81
6.3 Value of slope for each subtest 82
6.4 Ranking of reliability and linear regression 83

List of abbreviations

PdM Predictive maintenance

DUT Device under test

RUL Remaining Useful Lifetime

ML Machine Learning

DMM Digital Multimeter

DB Database

HW Hardware

SW Software

ETL Extract Transform Load

1 Introduction

Maintenance is a crucial task in the industry which highly relies on the productivity and availability of its systems. Since any unplanned failures and downtimes of production lines negatively impacts a company's business, maintenance has a strong influence on a company's competitiveness in low pricing, high quality, productivity and performance. Any misjudgements in choosing a proper maintenance system can result in significant financial as well as reputational losses.
According to a market study by Panemon Institute, organizations lose $138,000 per hour due to data centre downtimes [11]. Another report shows that the Operation and Maintenance costs for offshore wind turbines amounts to 20% to 35% of the total revenue of the generated electric energy [17]. It is also reported that the maintenance expenditures in oil and gas industry ranges from 15% to 70% from total production costs [8][29]. Therefore, it is imperative for companies to acquire a suitable and efficient maintenance strategy.
Through evolution of modern technologies such as Internet of Things, sensor technology, machine learning, etc. the state of the art for maintenance strategies has been shifting from *Reactive Maintenance* to *Preventive Maintenance* and finally to the *Predictive Maintenance* . Although the early history of predictive maintenance is not formally documented for its early days, it has been utilized in the industrial world since the 1990s as part of the condition-based maintenance.
The perks of this maintenance strategy are its ability to predict a failure ahead of time and determine potential anomaly. These characteristics offer many benefits if it is properly applied which is part of its rising popularity in the last decades.

Since predictive maintenance (PdM) is a broad topic, the following sections give a rough introduction into the term predictive maintenance in general, some of its advantages and disadvantages as well as requirements that need to be met in order to be able to implement a profitable PdM.

1.1 What is predictive maintenance?

Maintenance, in general, can be described as an effort to keep the condition and the performance of a system equal to its normal state. The maintenance activities can be categorized into the following types: planned and unplanned maintenance activity.

Planned maintenance is timely preplanned and carried out according to that plan whereas **unplanned maintenance** relies on different methods to determine the

timing of the maintenance.
Different systems require different **maintenance strategies** depending on the type of system, costs and criticality levels but the following types of maintenance are most commonly used:
Corrective maintenance, *Predetermined maintenance*, *Predictive maintenance*
The hierarchy of the maintenance strategies is shown in figure 1.1. It shows the different strategies and its hierarchical dependencies. As opposed to the different

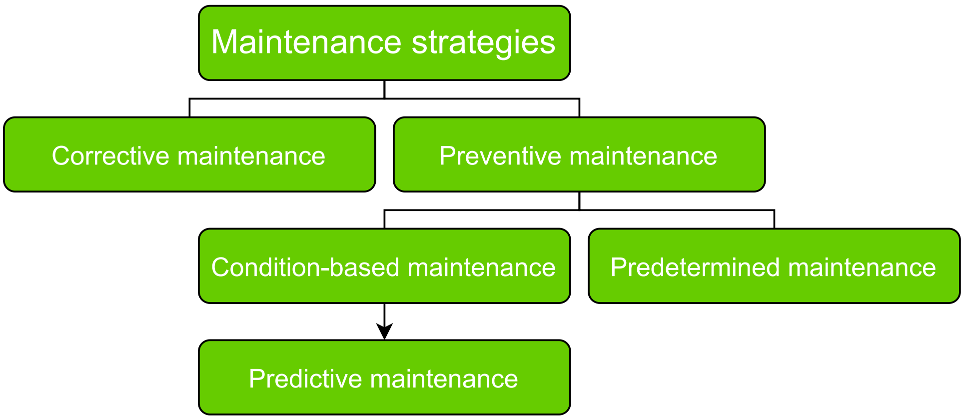


Figure 1.1: Maintenance strategies (own illustration, based on [32][42])

types of maintenance predictive maintenance is a proactive maintenance technique which aims to reduce breakdowns of a system by predicting a possible failure. It uses historical data of the system and advanced analytics to estimate when a breakdown will occur. PdM techniques can also include more modern technologies such as machine learning techniques and artificial intelligence. Depending on the PdM forecasts a proper maintenance can be scheduled in advance before the system can experience a downtime.

1.2 Why to use predictive maintenance?

Predictive maintenance systems offer many advantages that can often be very attractive.

1.2.1 Benefits of PdM

Assuming a proper maintenance strategy including proper techniques have been selected, a PdM could yield the following advantages:

- **Optimized scheduling:** Since the PdM can forecast the time of required maintenance, the activities can be scheduled at the most optimal timing.

- **Reduced maintenance costs:** As opposed to other maintenance techniques when maintenance is carried out even when it is not required, predictive maintenance eliminates such inefficiencies.
- **Improved safety:** PdM can alert regarding any imminent system failures if the PdM strategy allows anomaly and fault detection.
- **Increased productivity:** Preventive maintenance strategies generally have the advantage that the production chains are not disrupted by systems failures. Thus, generally, the entire workflow of the production is never interrupted. A predictive maintenance strategy further optimizes on the timing of the maintenance.

1.2.2 Drawbacks of PdM

Even though a PdM might seem to be a very attractive tool, one should be aware of some potential drawbacks of PdM under certain circumstances which are listed below.

- **Large upfront cost:** This applies for all strategies but is even more true for PdM since it requires condition monitoring tools, analytical software tools, trained employees, man hours for the setup of the entire PdM system and potentially other costs.
- **Required expertise:** Trained employees are needed to use the PdM system, such as using the monitoring equipment, interpreting data and analyzing reports from the PdM system.
- **Not cost-effective for all types of systems:** For certain system, such as facility-centric environment, other maintenance techniques are often cheaper and more effective. It is not profitable setting up a PdM system for a low-value asset.
- **Not suited for all systems:** It is not suited for systems with random failure modes or without initial data to predict failure. A random failure mode implies failures which occur randomly and therefore, no type of preventive maintenance can be applied.

1.3 When to use predictive maintenance?

Following the advantages and the disadvantages for PdM systems some rough guidelines can be concluded which help in choosing a proper maintenance method.

1.3.1 Use it!

If at least one of the following points is satisfied the predictive maintenance method could be suitable:

(1) high system cost

(2) great system criticality

(3) high repair and replacement costs

Even if these points are fulfilled, it is important to ponder whether the use of a PdM system will yield a favourable return on investment (ROI). This means that the saved money used on a reduction in system failures should meaningfully exceed the costs of maintenance.

1.3.2 Don't use it!

Some possible red flags for not implementing a PdM are the following:

(1) "Wrong/Not suitable" type of system

(2) system is cheap

A "wrong" type of system would be e.g. a roof of a building which has to be maintained, meaning possible leaks have to be detected. A predictive approach would require too many sensors leading to a high cost. In this case a preventive method (e.g. periodic inspections) would be a more appropriate option.
If the system or its maintenance is cheap, it is often times not profitable to use a PdM because of the large upfront cost for PdM. In such a case it is a better solution to resort to corrective maintenance where maintenance is only performed when a breakdown or failure occurs.

1.3.3 Conclusion

Generally, one can try to implement a PdM if the above-mentioned requirements are partially satisfied. The U.S. Department of Energy's Federal Energy Management suggests in its "Operations and Maintenance (O&M) Best Practices Guide" to use PdM services from outside vendors and rely on their equipment and expertise if a serious commitment to proper implementation, PdM operator training and equipment monitoring and repair cannot be made [15].
One way to facilitate the first use of a PdM would be the utilization of *Computerized Maintenance Management Systems* (CMMS). A CMMS is a software solution that centralizes maintenance information, facilitates processes, and automates some tasks to improve efficiency. Such a software would remove the need to implement PdM manually, but trained personnel would still be essential.

1.4 Requirements for PdM

1.4.1 Suitability of systems for PdM

Some of the previous arguments for or against PdM contain the argument of using "correct/suitable" types of systems. It is difficult to categorize such systems but the above-mentioned "Best Practices Guide" contains the table seen in figure 1.2. Those are some common predictive technology applications or types of systems that are used for PdM. These systems can be considered as the correct type of systems and suitable for PdM because they have been thoroughly analyzed and their suitability has been shown.
Generally, when implementing a PdM for a commonly used system as shown in figure 1.2 it is well researched which signal/sensors and which analysis type to choose [15]. If that is not the case it is important to understand the given system, its potential critical faults and which signals indicate which failure mode. Those signals with the capability to capture a certain fault are actively used in a PdM system and are called *condition indicators*.

Technologies / Applications	Pumps	Electric Motors	Diesel Generators	Condensers	Heavy Equipment/ Cranes	Circuit Breakers	Valves	Heat Exchangers	Electrical Systems	Transformers	Tanks, Piping
Vibration Monitoring/Analysis	X	X	X		X						
Lubricant, Fuel Analysis	X	X	X		X					X	
Wear Particle Analysis	X	X	X		X						
Bearing, Temperature/Analysis	X	X	X		X						
Performance Monitoring	X	X	X	X				X		X	
Ultrasonic Noise Detection	X	X	X	X			X	X		X	
Ultrasonic Flow	X			X			X	X			
Infrared Thermography	X	X	X	X	X	X	X	X	X	X	
Non-destructive Testing (Thickness)				X				X			X
Visual Inspection	X	X	X	X	X	X	X	X	X	X	X
Insulation Resistance		X	X			X			X	X	
Motor Current Signature Analysis		X									
Motor Circuit Analysis		X				X			X		
Polarization Index		X	X						X		
Electrical Monitoring									X	X	

Figure 1.2: Common predictive technology applications [15]

1.4.2 Requirements for datasets

The datasets of a system have to be analyzed in its suitability for a PdM. This analysis contains the following steps [25]:

Identification of data sources: The available data sources have to be identified first. It is important to have multiple data source, such as product data, maintenance data, machine data, etc.

Quality of data: Moreover, the quality of data has to be checked and evaluated which is a requirement in order to be able to perform analyzes later on. Some features that indicate the quality of data are completeness, granularity, size of data, time stamps, etc.

Use cases: The analysis of the quality of the data can yield an insight whether there are any PdM related uses cases for the given system which are shown in figure 1.2. If a suitable use case or technology is found, the PdM approach is more likely to succeed. Otherwise, condition indicators have to be found and a PdM has to be designed and implemented manually.

1.5 Workflow of PdM

Since a PdM system is a data-driven system, it has to be trained using the historical data of the system. A rough workflow for this process can be seen in figure 1.3.
The first step is to gather the data from the sensors which are refined into more useful datasets in the preprocessing step. When developing the prediction model, it is imperative to select proper condition indicators for the machine learning model. Based on those condition indicators the model for predicting the Remaining Useful Life (RUL) is trained. A PdM strategy can also contain an anomaly or fault detection model which requires the same steps as shown for the RUL model.
The selection of the condition indicators is an iterative process which often requires a trial-and-error approach to determine the ideal condition indicators and the most suitable machine learning algorithm.

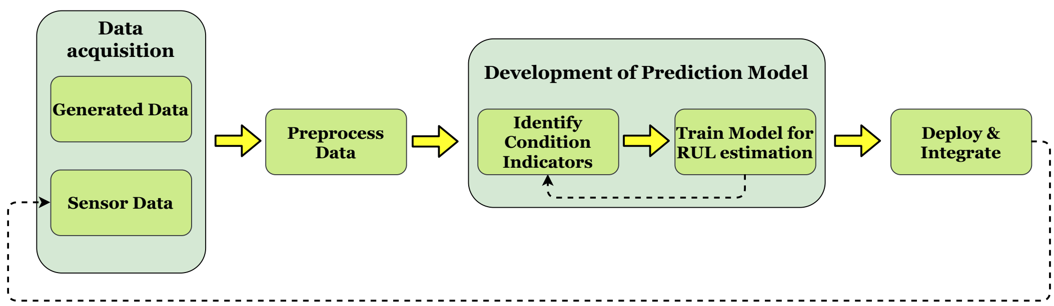


Figure 1.3: PdM workflow

Depending on the available data the type of RUL prediction can vary. Some of the different dataset types are the following:

Lifetime data is a dataset which indicates how long it took for a similar system to reach failure. If lifetime data is available, proportional hazard models and probability distributions of component failure times can be used to estimate the RUL [41]. A simple example would be the estimation of the discharge time of a battery base on past discharge times.
The survival function plot in figure 1.4 shows the probability that a battery will fail based on how long it has been in operation. The plot shows, for example that if the battery is in operation for 75 cycles, it has a 90% chance of being at the end of its lifetime.

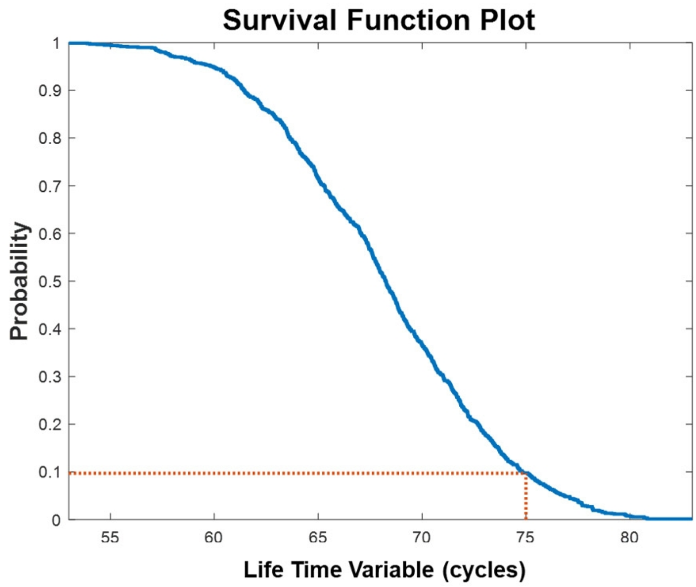


Figure 1.4: Lifetime data

Run-to-Failure data datasets of similar components or different components with similar behaviour can be used to estimate the RUL using similarity methods. These methods analyze the degradation profiles and compare those with the incoming new data points. The points are matched with the profile that they are the most similar with [41].
In Figure 1.5 the degradation profiles of historical run-to-failure data sets from an engine are shown in blue and the current data from the engine is shown in red. Based on the profile the engine most closely matches, the RUL is estimated to be around 65 cycles because the distribution of the stars (or endpoints) of the nearest blue curves gives an RUL of 65 cycles.

Threshold data is a known value for a condition indicator. A degradation model estimates the RUL by predicting when the condition indicator will cross the threshold. Figure 1.6 shows an exponential degradation model that tracks failure in a high-speed bearing used in a wind turbine. The condition indicator is shown in blue.

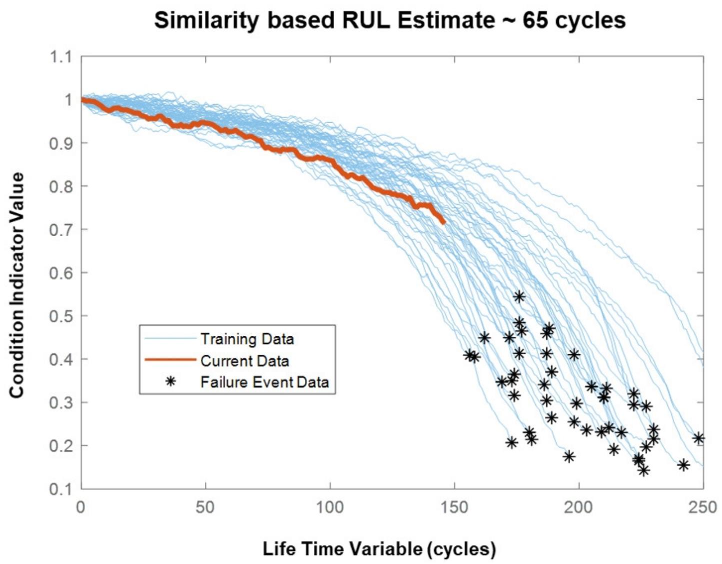


Figure 1.5: Run-to-failure data

The degradation model predicts that the bearing will cross the threshold value in approximately 9.5 days. The region shaded in red represents the confidence bounds for this prediction.

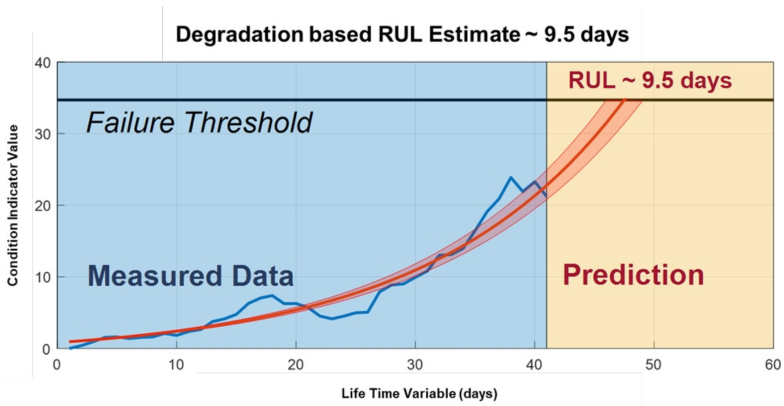


Figure 1.6: Threshold data

By taking the above-mentioned dataset types into consideration, a prediction model can be selected. Once the prediction model is implemented, the predicted RUL can be used for alarm and maintenance systems. New data points are continuously fed into the data gathering process, consequently closing the loop as shown in figure 1.3.

2 Research question

The company OMICRON electronics GmbH wants to evaluate its own systems on the suitability for PdM. It uses test towers for testing its devices and stores the unit test results in a database. Even though there are other OMICRON devices which also gather data, the test towers are the main focus of this research. Since the test towers are the main testing place of OMICRON devices it is of great interest to know if it is possible to predict a failure or an anomaly for the test towers. Such a maintenance strategy would have many benefits for the company as listed in section 1.2.1.
Thus, the following question arises which is to be answered in the context of this research.

"Is it possible to implement a PdM for a test tower?"

The research question above implies some additional questions.
What kind of condition indicators are needed or are best suited to detect and analyze trends? How suitable is the data quality? Are there any trends in the data which can be used? Can a RUL be predicted? How accurately can the RUL be predicted? Is it also possible to detect anomalies?

The **ideal solution** would be the following.
Firstly, detect which subtest will fail or show a measurement result which lies beyond the specifications. Additionally, the exact RUL can be acquired. Moreover, the exact location of the error location can also obtained thus enabling fast and efficient maintenance procedures.

The following chapter gives an overview of the hardware and software setup as well as a rough explanation of the unit test.

3 System setup

3.1 Test tower

The test tower at OMICRON electronics GmbH is used to perform unit tests inside a climate chamber for company-specific devices, such as the CMC430. A possible test setup with the test tower is shown in figure 3.1. The test tower has access to multiple servers shown as blue server icons.

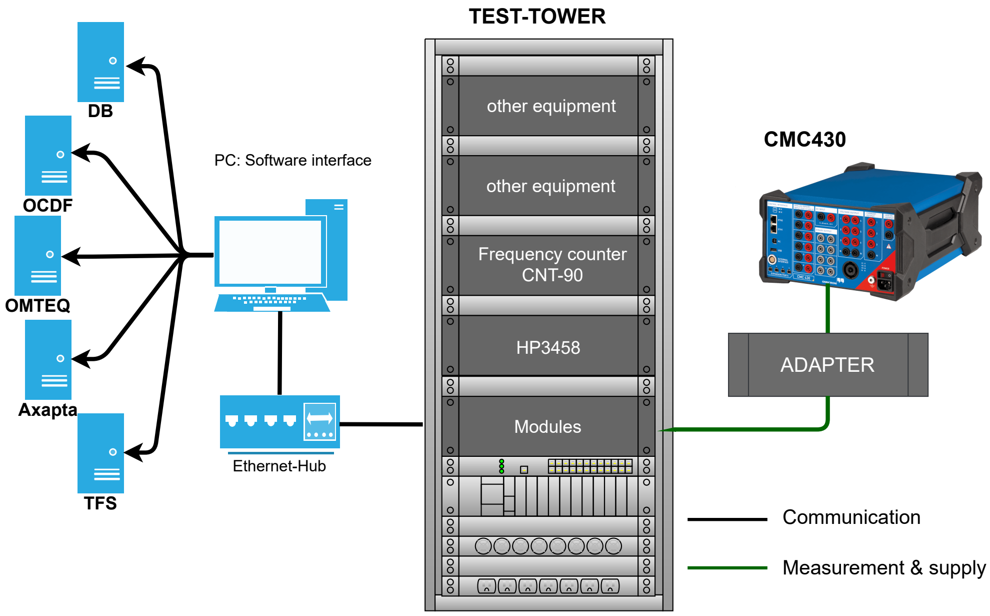


Figure 3.1: Test tower setup

DB	= Database
OCDF	= Omicron Calibration Data Frontend
OMTEQ	= Omicron Measurement Equipment
Axapta	= Axapta, object-oriented ERP system
TFS	= Team Foundation Server
*CMC*430	= Device under test (DUT), an OMICRON device

Figure 3.2 shows an actual test tower setup in OMICRON along with some labels. It shows the same setup as it was shown abstractly in the previous figure where the CMC430 DUT is connected to the tower through an adapter. The location of the test

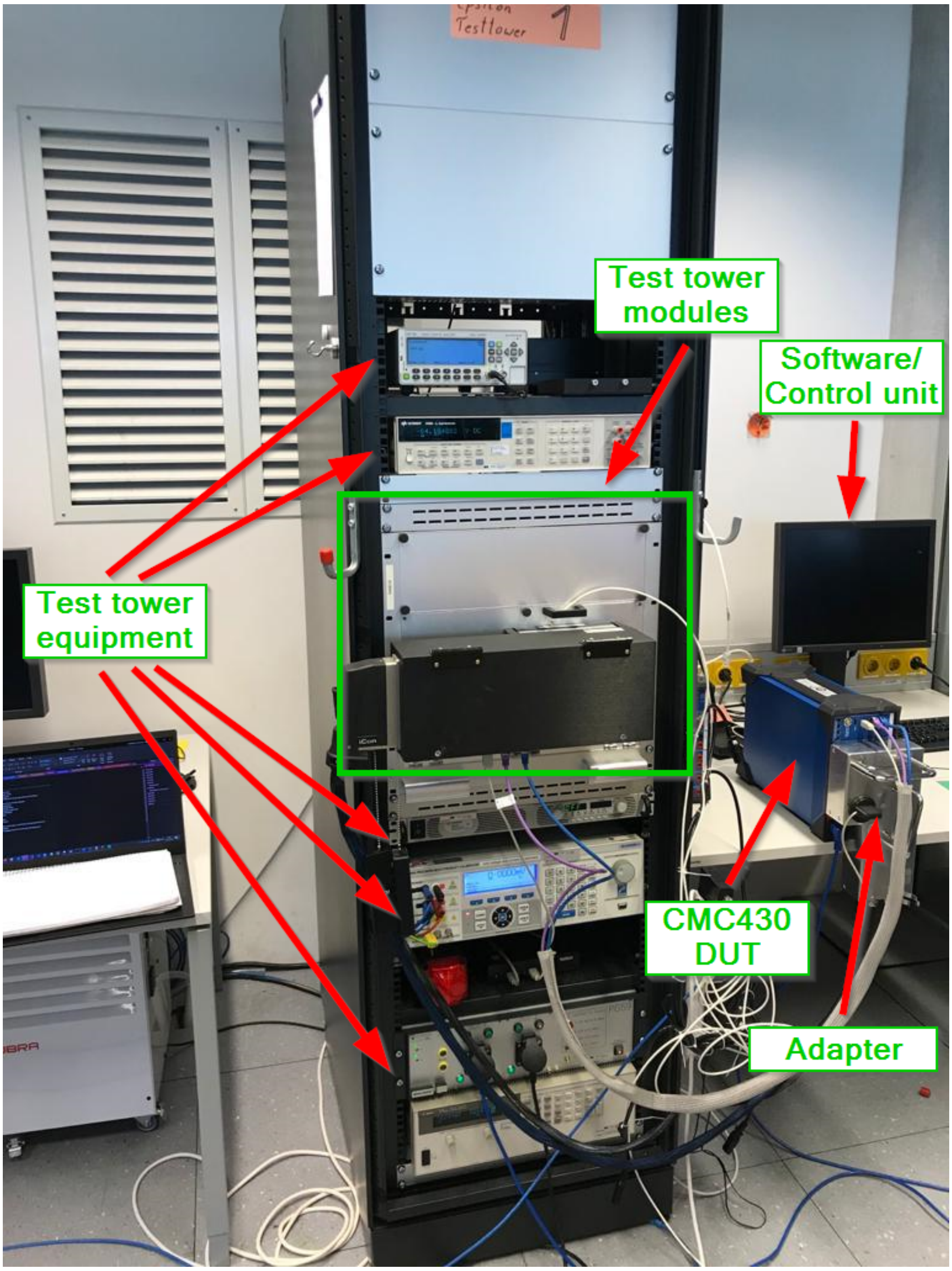


Figure 3.2: Actual test tower setup

tower modules and the test equipment is shown as well.
The entire test tower setup consists of the components described in table 3.1.

Tower components	Description
Test tower equipment	e.g. DMM HP3458
Test tower modules	-
DUT adapter	-
DUT	e.g. CMC430
Control unit	see PC + software + DBs

Table 3.1: Test tower components

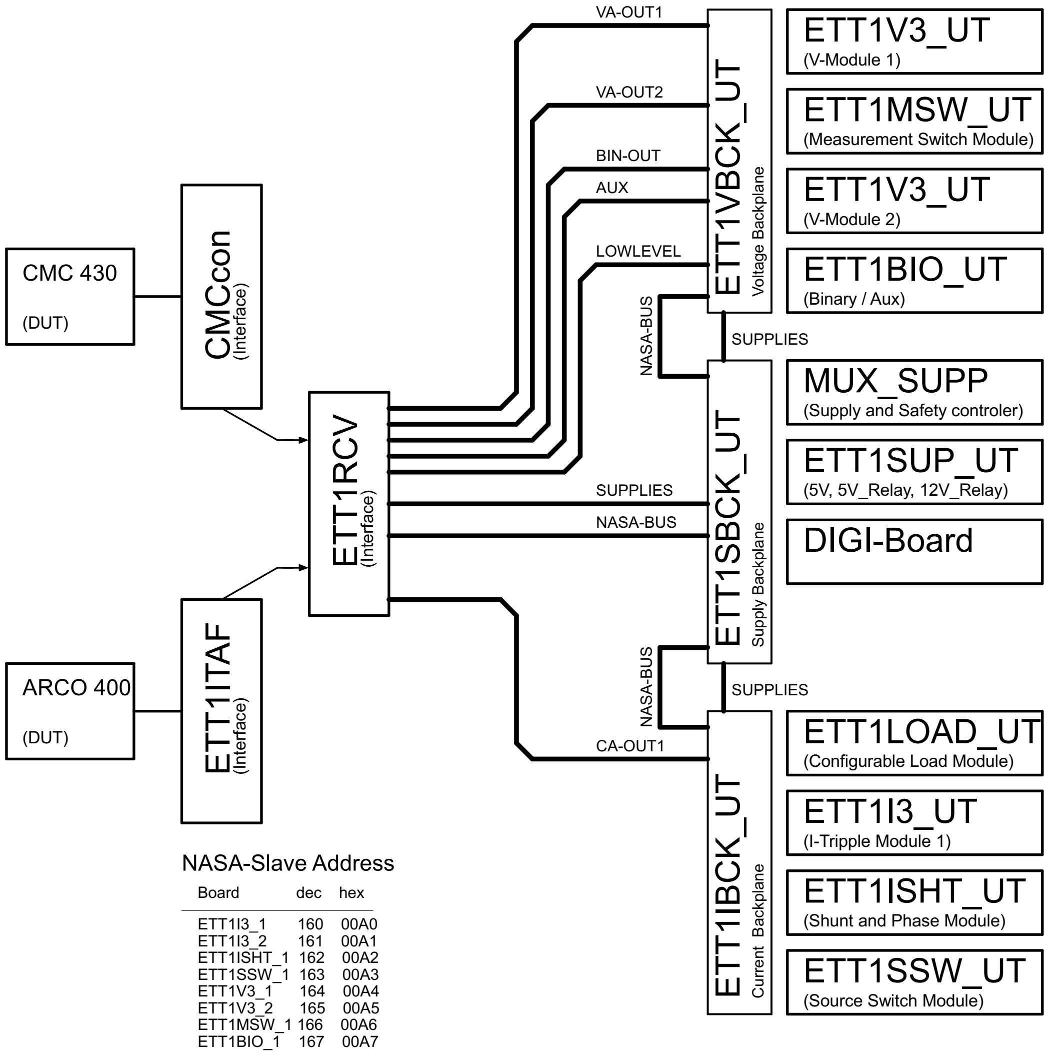


Figure 3.3: Modules of the test tower

The test tower equipments are devices such as DMMs, sources, frequency counters and other measurement and stimulating devices. The test tower modules have essentially the tasks to route the signals from the test tower equipments to the DUT. These modules mainly consist of relays but also contain other component such as shunts. Nevertheless, the main focus of this research are on the relays. The assumption is that the relays are the main source of errors in the test tower.
The modules of the test tower can be seen in more detail in figure 3.3. Every block on the right hand side is a specific module dedicated to a certain type of task.

There are a total of two test towers in the climate chamber which can be analyzed, called "ETT1" and "ETT2". Both test towers a constructed equally and use the same software for operation.

3.2 DUTs

There are two different DUTs, which are tested on the test towers. Those DUTs are CMC430 and ARCO400 which are shown in figure 3.4 and figure in 3.5.

Figure 3.4: CMC430, an ultra-portable Protection Test Set and Calibrator [26][10]

Figure 3.5: ARCO400, a universal test set for recloser controls[27] [7]

This research focuses only on the CMC430 device. The CMC430 itself consists of several submodules. For this research the most important ones are listed in table 3.2.

Module	Description
RST1VA	Voltage amplifier
RST1CA	Current amplifier
AFE400	Analog input
SWT1AFE	DC input

Table 3.2: Relevant CMC430 modules in the context of this research

The tests for these subtests contain the effects of the test tower in their measurements which is crucial for this research. This is explained further in the next section.

3.3 Unit test

The unit test of the CMC430 with the test towers has a hierarchical structure. The first level is shown in figure 3.6. Each of those test points contains several subtests.

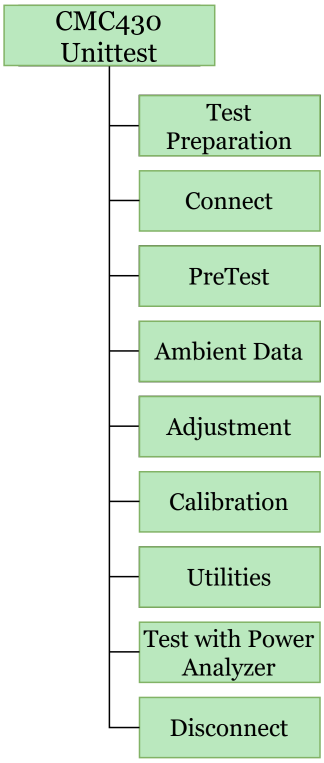


Figure 3.6: CMC430 unit test structure - level 1

Most of the shown test points do not measure any signals using the relay paths of the test tower apart from the **adjustment** and the **calibration** test points. Those relay signal paths are important because they are the paths which include any possible effects of the test tower on the measurements.

In the context of this research however, only the adjustment test point is of interest. The adjustment subtests determine the offset or error between the DUT and the rest of the measurement setup including the tower itself. In the next test point, which is calibration, the actual measurements are performed by considering the previously measured offsets. This means that the effects of the signal paths are removed in the calibration test point. This type of 2-step-measurement results in a

high accuracy calibration of the DUT.
Since this research is analyzing the tower itself, the calibration test point is therefore not of much importance and only the adjustment test point is evaluated.
Each subtest is repeated using different parameters, such as different voltage ranges. Additionally, each repeated subtests have to be repeated once again for each channel of a module if that module contains multiple channels. The RST1VA and the RST1CA modules have 6 and 3 channels, respectively.
In the context of this research the channels are treated as one channel and only one range is selected for each subtest. Furthermore, some subtests, which have been deemed useless for PdM, are not included. More details on these decisions are thoroughly explained in section 5.6.

3.4 Database

All the measurement results for the unit test are stored in a database as seen in figure 3.1. Those datasets can be accessed by SQL queries and used for analytics.

3.5 Software

Python is the primary tool for various data analytics tasks, machine learning model implementations and for general programming in this research. Some reasonings are the multitude of data mining, data analytics and machine learning tools which are available for free for python. Furthermore, there is previous experience with the programming language.

4 Modelling of test tower

To get an understanding for the structure of each subtest and the test tower in general, the first approach is model-driven. Each subtest is analyzed on the types of components and the number of components it has. This information is then used to perform a reliability analysis.
The reliability analysis result is later used to cross-compare with the data analysis result.

4.1 Limitations

The test tower mainly consists of relays. Even though there have been maintenance activities with changing relays, the greater majority of the relays remained untouched which could potentially fail in the future. Other components are being ignored in the model. There are two main reasons:

1. Some components such as shunts and loads, **do not remain in the same test tower**. Meaning, each test tower uses shunts, but after maintenance activities it is not guaranteed that the exact same shunt remains in the tower.

2. The reliability **analysis requires failure rates** for the components but shunts and loads do not provide such information or it is hard to acquire that information.

4.2 Distribution of relays in subtests

For the tower modelling two test points are analyzed - adjustment and calibration which are depicted in figure 3.6. These test points are those, which contain the majority of the relay usages. By analyzing the electric schematic of the test tower, the exact number of relays for each subtest signal path and the exact types have been extracted. The entire test tower has approximately 250 relays whereas in the adjustment test point only 66 are actively used and in the calibration test point 129. In order to account for the ARCO 400 DUT which is also tested on the same test towers, the number of used relays can be doubled.

Figure 4.1 shows the relays used for different subtests. This figure is only a section of the entire matrix. Each line represents a subtest and the colors and numbers correspond to a relay type which is shown in table 4.1. Using this spreadsheet, it is possible to get the exact number of used relays and the relay types for each subtest

Relay type	Index
None/not used	0
IM02G	1
NA5W	2
DSP2A	3
PICK_104_1_A	4
PICK_104_2_A	5
ALFG2PF12	6
G2RL_24	7
PB124012	8

Table 4.1: Relay types and their corresponding indices

for the CMC430. The x-axis shows the names of the relays which corresponds to a specific module in the tower.

The tower uses different relay types which are listed in table 4.2 along with the electrical and mechanical endurances. These endurances are equivalent to the MBTF (Mean Time Between Failure).

Relay Type	Electrical endurance [in Operations]	Mechanical endurance [in Operations]
IM02G [19]	1×10^5	1×10^8
NA5W [24]	5×10^5	1×10^8
DSP2A [13]	1×10^5	5×10^7
PICK_104_2_A [2]	1×10^6	1×10^8
PICK_104_1_A [1]	1×10^6	1×10^8
ALFG2PF12 [5]	3×10^4	1×10^6
G2RL_24 [16]	5×10^4	2×10^7
PB124012 [6]	1×10^6	5×10^6

Table 4.2: Used relay types in test tower

The mechanical and electrical endurance values describe different characteristics and should not be compared. The failure criteria for the mechanical endurance are not the same as for electrical endurance, therefore the value for mechanical endurance has no relation and cannot be directly compared to the electrical endurance for very low loads[12]. The main difference between these endurances lie in the used loads during testing.
The **mechanical endurance** describes the number of cycles or switching operations of the relay without load during which the relay remains within the specified characteristics. The **electrical endurance** is defined as the number of operations until switching failure of a relay under defined conditions of load and ambient. The specified reference values for the life of the relay apply, within a resistive load window. At lower contact loads a substantially longer electrical life is achieved.

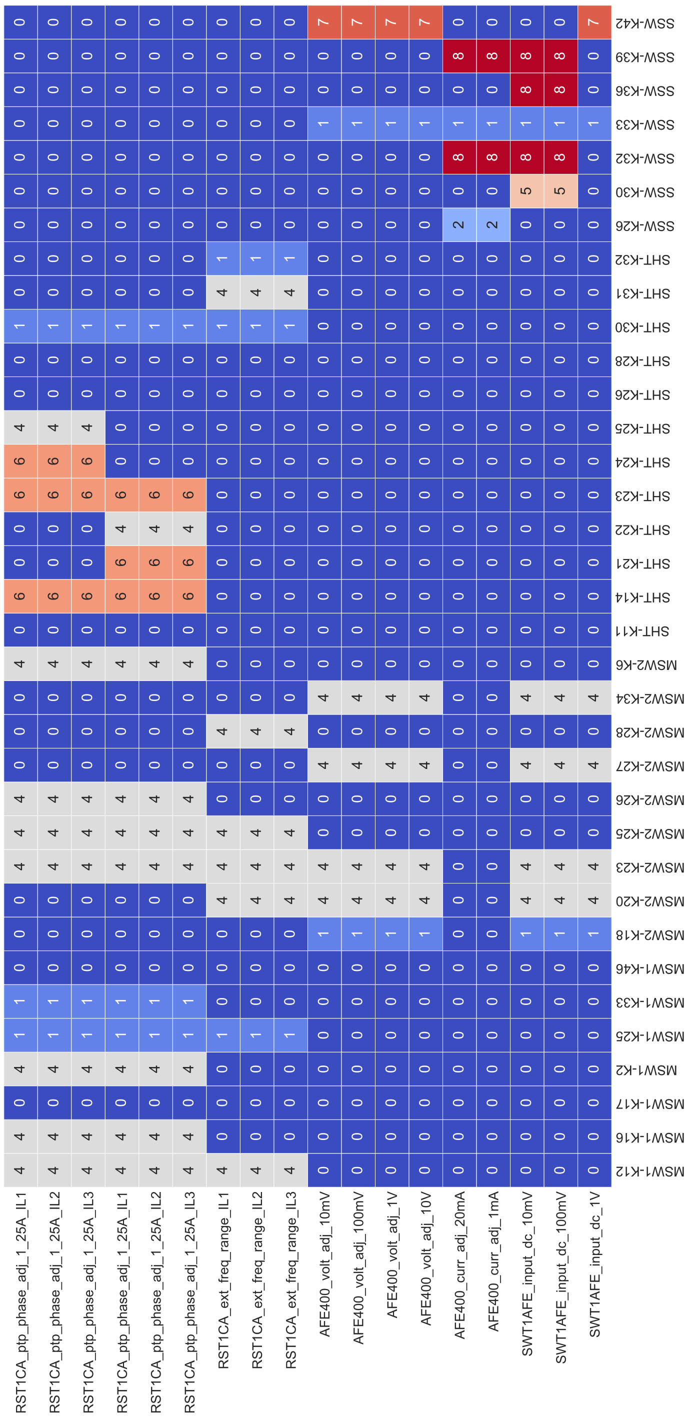


Figure 4.1: Used relays for subtests in CMC430

At higher loads the electrical life is reduced substantially [30]. The relays from AXICOM for example have the following definition for a failure during electrical endurance test[46]:

- contact resistance value higher a certain limit ($> 1\,\Omega$)
- insulation resistance less a certain limit ($<100\,\text{k}\Omega$)
- bridging resistance lower than a certain limit ($<100\,\text{k}\Omega$)

The **distribution of each relay type** for both test points looks similar. The figure 4.2 shows the distribution of relays for the calibration test point.

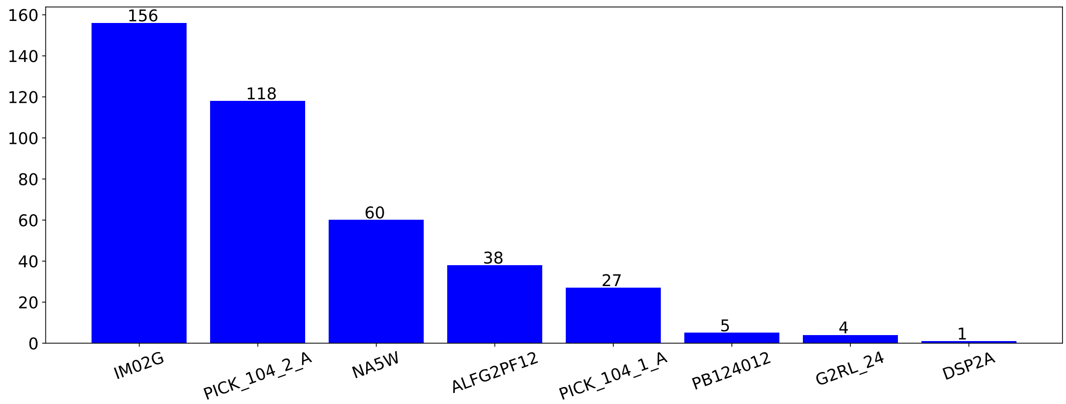


Figure 4.2: Distribution of relay types

Since the exact number of relays for each subtest is known along with the exact type for each relay, a reliability modelling can be performed. The process of modelling a system with multiple components is explained next.

4.3 Reliability analysis

The reliability of one single component can be modelled with the equation 4.1.

$$R(t) = e^{-\lambda t} \tag{4.1}$$

where:

R = Reliability
λ = Failure rate
t = Time or number of operations

The failure rate λ can be calculated if the $MTBF$ of the component is known with the equation 4.2.

$$\lambda = \frac{1}{MTBF} \tag{4.2}$$

In case of relays the MTBF is measured in number of operations instead of a time unit. There are different methods to combine multiple components. The components can either be in parallel or in series.

From reliability point of view a **series system** is such that fails, if any of its components fails. A **parallel system** is such that it only fails if all its components fail [14]. The test tower has to be modelled as a series system since all its relays have to function properly in order to pass the unit test and none of the relays are used as redundance. Thus, assuming a series configuration, the cumulative reliability for the tower can be calculated with the equation 4.3.

$$R_{tower} = \prod_{i=1}^{N} R_i(t) = \prod_{i=1}^{N} e^{-\lambda_i t} \tag{4.3}$$

where:

N = Number of components
i = i-th component

4.4 Simulation results for the test tower

The simulation results for the entire tower are seen in figures 4.3 and 4.4. Figure 4.5 shows the comparison between the mechanical and electrical endurance The x-axis shows the number of runs. The current time point can be calculated by finding the number of test runs that were performed up to the current time for the CMC430 and the ARCO 400. The current time point is shown as the red dashed vertical line. The electrical endurance has a reliability of 0% for the current time point.

The mechanical endurance is stable even after 4.2 years at around 97%. Generally, an acceptable reliability is a value above 80%.
For both the mechanical and electrical endurance a failure is not a failure per se but an exceedance of the specifications. Since this effect does increase the probability of a failure of the subtests, this modelling is a valid approach in modelling the test tower. But a reliability value of 0% should not be considered a total breakdown of the system but rather a high probability of an increased error in the system.

The reason why the result of the electrical endurance reliability is so low is because many NA5W relays are used and each of those have a relatively low electrical endurance. As a result, this value propagates through the formula and results in a low overall electrical reliability.
Since not all subtests perform test with loads, there are differences in the remaining reliability for different subtests. This effect is not analyzed any further for each subtest

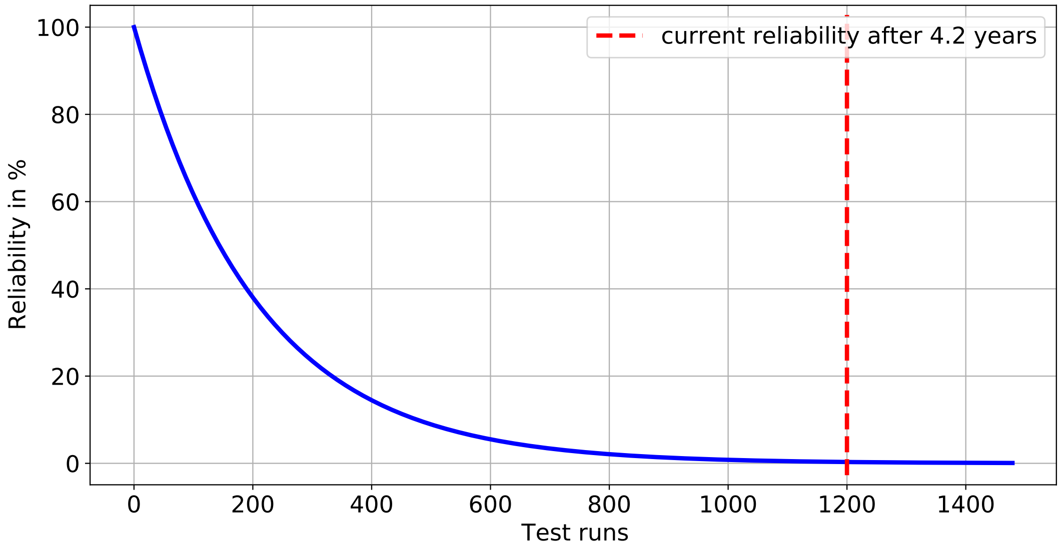


Figure 4.3: Reliability for electrical endurance

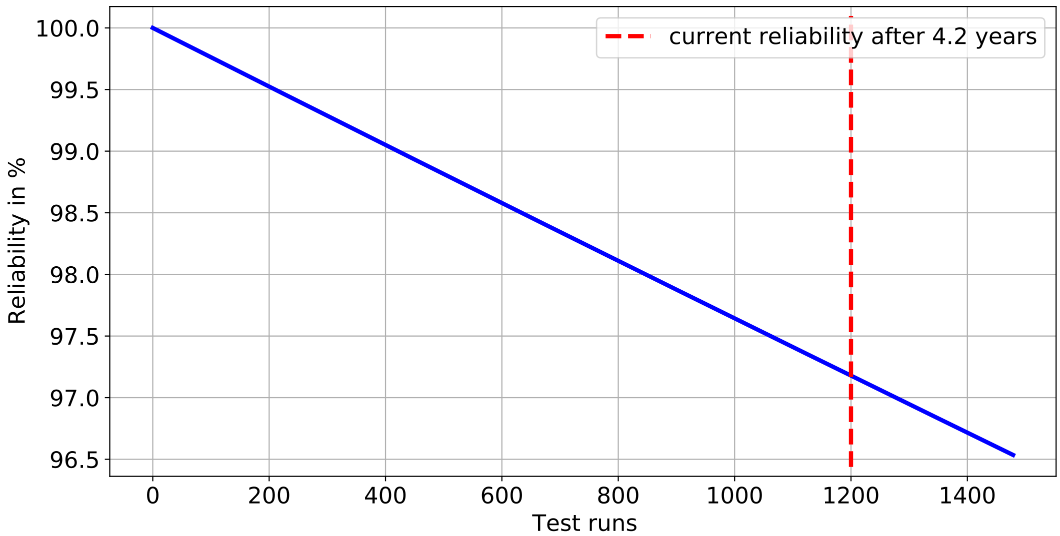


Figure 4.4: Reliability for mechanical endurance

because the main takeaway is the ranking of the subtests as it is explained in the coming sections.

The NA5W relay has a relatively low electrical endurance compared to other relay types and therefore an alternative might be considered.
The ALFG2 relay has a low endurance for both endurance types. But these relays has been widely replaces by E-relays resulting in a better endurance. Therefore, the

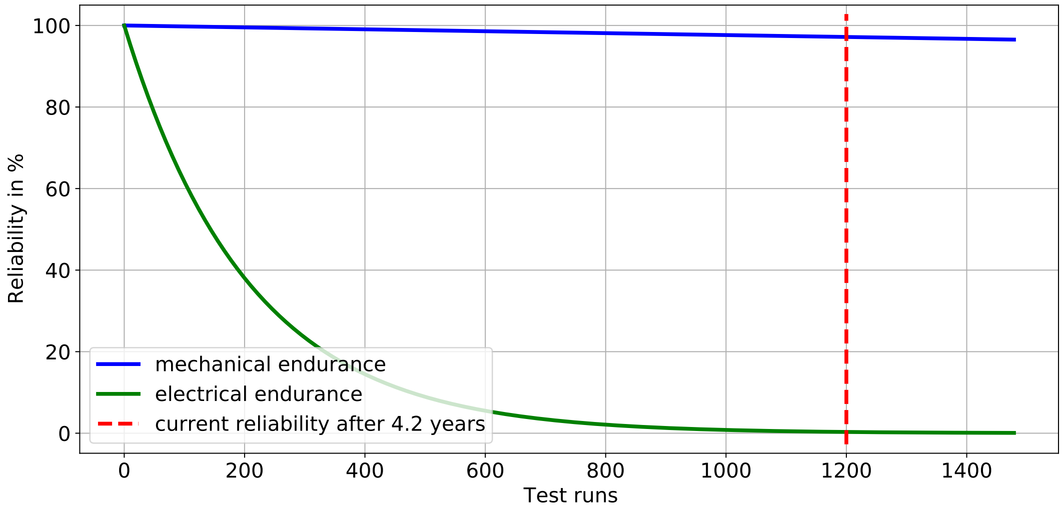


Figure 4.5: Comparison of mechanical and electrical endurance

ALFG2 relays have been ignored in the modelling process shown above by setting their reliability to 100%.

4.5 Simulation results for separate subtests

It can be observed that the reliability of a subtest is proportional to the number of relays used in that subtests. Therefore, since the number of relays for each subtest is known, it is possible to calculate the reliability of each subtest.
By ranking the subtests by their number of relays results in a list of subtests sorted by their reliability. This list is later compared with the data analysis results of the subtests in chapter 6.4.
The hypothesis is that the effect of the relays is big enough that it can be detected as a deterioration effect in the dataset.

5 Data preparation

5.1 Selected PdM model

It can be observed that the given data is a threshold data as explained in chapter 1.5. Thus, the main goal is to **find a trend in data** which will enable the use of the **degradation model** mention in chapter 1.5.

5.2 Requirements for final dataset

The goal is to have a dataset with a **relatively small dimensionality** by removing redundant data in order to decrease potentially long training times of ML models. Furthermore, since it is already known that the unit test has many subtests, a **manageable number of subtests** is needed. This can be achieved by removing redundant subtests in the dataset.

5.3 General information

The unit test of the CMC430 consists of a hierarchically structured test with a multitude of subtests. The subtests use the following characteristics when testing:

- **Functionality:** A specific subtest tests a specific functionality of the CMC430.
- **Parameters:** Each subtest is run multiple times using different parameters to get a more thourough understanding of the functionality.
- **Channels:** Each subtest with its different parameters is also run on different channels if any exist.
- **Types of measurement results:** Some tests yield multiple results which are different in units e.g. voltage + phase measurement

Due to testing each functionality and each channel with many different parameters the unit test has many test points and all those test points yield a number of data points that all have to be analyzed in the context of this thesis in some shape of form.

5.4 Data presentation

The table in 5.1 shows all the different subtests with its characteristics for the unit test of CMC430.

Module	Subtest	Number of Parameters	Channels	Types of Result Units	Sum of Subtest Runs
RST1VA	High Range (150V)	7	6	1	42
	PTP Phase Adjustment High Range	1	6	2	12
	Extended Frequency Range	1[a]	6	2	12
RST1CA	Low Range DC (1.25 A)	8	3	1	24
	High Range DC (12.5 A)	8	3	1	24
	Low Range AC Gain Burden (1.25 A)	6	3	2	36
	High Range AC Gain Burden (12.5 A)	4	3	2	24
	PTP Phase Adjustment (1.25 A) @ 55 Hz with load	3	3	2	18
	PTP Phase Adjustment (12.5 A) @ 55 Hz with load	4	3	2	24
	Extended Frequency Range	1[b]	3	2	6
AFE400	Voltage DC Adjustment	8	1	1	8
	Current DC Adjustment	4	1	1	4
SWT1AFE	Input DC Adjustment	12	1	1	12
	Input AC Adjustment	12	1	1	12
	Adjust Phase between Ranges	1	1	1	1
				Sum of subtests:	259

Table 5.1: Summary of subtests for CMC430

[a] see explanation below

[b] The subtest has 14 different harmonics as parameters which range from the 9th to the 56th harmonic. In this research only the 9th harmonic is analyzed in more detail. The interpretation of harmonic measurement results and their development over time as well as their comparison is rather difficult to interpret. Thus, only one harmonic, namely the 9th harmonic, is analyzed.

The table shows only the subtests for the "adjustment" part from the hierarchy shown in figure 3.6. Some parameters and subtests contain measurements where the test tower is not used, such as initializations etc. Such tests do not contain any relevant information for this thesis and are therefore ejected. The exact hierarchy of the subtests can be seen in the appendix in 9.
Since the main focus of this thesis is the data analysis process, the exact names of the modules, subtests, parameters etc. are removed and replaced by aliases which can be seen in table 5.2. Henceforth, the modules and subtests are called by their aliases.
The aliases for the parameters can be seen in table 5.3. Not all modules are discussed in the context of this research which is explained in section 5.8.1. Therefore, only the parameters for the modules M1 and M2 receive aliases.
The channels for the modules M1 and M2 already have a numbering which is used as the alias, e.g. the channel VL1 for the M1 is simply channel 1 instead of VL1. The same goes for the other channels VL2-VL6 of the M1.

Module	**Module Alias**	**Subtest**	**Subtest Alias**
RST1VA	M1	High Range (150V)	Sub1
		PTP Phase Adjustment High Range	Sub2
		Extended Frequency Range	Sub3
RST1CA	M2	Low Range DC (1.25 A)	Sub1
		High Range DC (12.5 A)	Sub2
		Low Range AC Gain Burden (1.25 A)	Sub3
		High Range AC Gain Burden (12.5 A)	Sub4
		PTP Phase Adjustment (1.25 A) @ 55 Hz with load	Sub5
		PTP Phase Adjustment (12.5 A) @ 55 Hz with load	Sub6
		Extended Frequency Range	Sub7
AFE400	M3	Voltage DC Adjustment	Sub1
		Current DC Adjustment	Sub2
SWT1AFE	M4	Input DC Adjustment	Sub1
		Input AC Adjustment	Sub2
		Adjust Phase between Ranges	Sub3

Table 5.2: Aliases for modules and subtests

Module	Subtest	Parameter	Parameter alias
RST1VA	High Range (150V)	100V	A
		50V	B
		1 V	C
		0V	D
		-1V	E
		-50V	F
		-100V	G
	PTP Phase Adjustment High Range	-[1]	-
	PTP Phase Adjustment High Range	-	-
	Extended Frequency Range	-	-
RST1CA	Low Range DC (1.25 A)	1.25A	A
		0.25A	B
		0.1A	C
		0A	D
		-0.1A	E
		-0.25A	F
		-1.25A	G
	High Range DC (12.5 A)	5A	A
		0.25A	B
		0.1A	C
		0A	D
		-0.1A	E
		-0.25A	F
		-5A	G
	Low Range AC Gain Burden (1.25 A)	1.25A @ 0 Ohm	A
		1A @ 0 Ohm	B
		1A @ 0.5 Ohm	C
		1A @ 1 Ohm	D
		1A @ 2 Ohm	E
	High Range AC Gain Burden (12.5 A)	5A @ 0 Ohm	A
		5A @ 0.5 Ohm	B
		5A @ 1 Ohm	C
		5A @ 1.5 Ohm	D
	PTP Phase Adjustment (1.25 A) @ 55 Hz with load	1 A @ 0.4 Ohm	A
		1 A @ 1 Ohm	B
		1 A @ 2 Ohm	C
	PTP Phase Adjustment (12.5 A) @ 55 Hz with load	5 A @ 0.25 Ohm	A
		5 A @ 0.5 Ohm	B
		5 A @ 1 Ohm	C
		5 A @ 1.5 Ohm	D
	Extended Frequency Range	-	-

Table 5.3: Aliases for parameters

[1] This subtest does not contain any parameters or only one parameter is used. Thus, a distinction is not necessary. The same goes for the other subtest whose parameters are marked with the “-” sign.

5.5 Data fetching

All the unit test results are stored in a DB and can be fetched using SQL queries. The general workflow when fetching data is shown in figure 5.1.

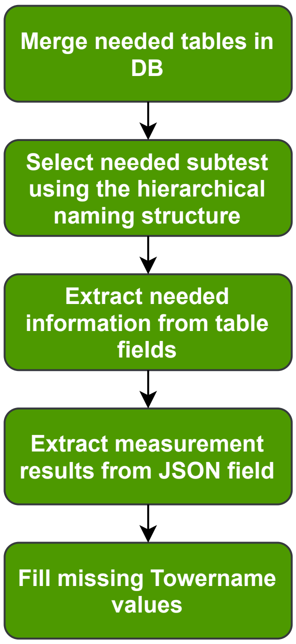


Figure 5.1: Subtest data from DB workflow

Merge needed tables in DB: Since the database is a relational database, all the information is logically divided into tables. In order to get the required data, several tables have to merged correctly.

Select needed subtest using the hierarchical naming structure: In this step the desired subtest is selected. Since the primary goal is to simply fetch the needed data from the DB, the string of the hierarchical naming of the subtest is used as the condition. A more efficient and "prettier" way to get the data would be to use the unique index keys which are assigned for all subtest items e.g. unit test, test run, measurement result, etc.
Listing 5.1 shows a snippet of an SQL query for fetching the subtest for the RST1VA High Range 150V subtest for the 0V parameter, or using the alias the M1Sub1 subtest with the parameter D. The hierarchical naming string as described above can be seen in line 9. The entire query can be found in the appendix in section 9.1.1.

```sql
SELECT
...

FROM
...

WHERE (tblTestResult.DeviceType='CMC430')
AND (_index.NodeName='0 V Adjustment')
AND (tblReportTestRunItem.Header='Adjustment  \  Voltage Amplifier  \
    High Range (150 V)')     -- condition to select subtest
AND(tblReportTestRunReport.TowerName = 'Epsilon01' OR
    tblReportTestRunReport.TowerName = 'Epsilon02' OR
    tblReportTestRunReport.TowerName is null)
AND len(_index.ComponentResultValuesSerialized) < 1000
AND (_index.ComponentResultValuesSerialized <> '[]')
AND (_index.ComponentResultValuesSerialized <> '[]')
AND (tblReportTestRunReport.Equipments <> '[]')
AND (tblReportTestRunReport.Equipments <> 'null')
AND len(tblReportTestRunReport.Equipments) > 1000
ORDER BY _index.ExecutionInfo_StartedAt
```

Listing 5.1: Example for fetching data for subtest using subtest name string

Extract needed information from table fields: Since not all attributes are relevant, only the following are selected:

- Serial number of currently tested DUT
- Tower name of the currently used test tower
- Measurement results can be one or more values (i.e. the measurement result is the error of the subtest. Sometimes that value is given directly as an absolute error and sometimes it has to be calculated using nominal and actual values)
- Timestamp
- Equipment list of currently used test tower

Extract measurement results from JSON field: The measurement results are given as a JSON structure. The individual data has to be extracted from the JSON data. In order to do so, the format of the JSON has to be known, as well as the names of the needed measurement result keys. The extraction is performed using available SQL functions.

Fill missing Tower name values: Not all tower names are available, some are missing due to a migration to a relational database in 2019. Therefore, all tower names before 2019 have to be filled by using the equipment list of the test tower. This is possible because some tower modules are unique for each test tower.

A raw dataset for a subtest with its current features has several disadvantageous characteristics which are amended in the following feature engineering process.

5.6 Feature engineering

5.6.1 Tower name feature

The first step is to select the data for a specific tower since there are two test towers available. As already mentioned not all tower names are available thus the missing ones have to be filled in. The table field with the equipment list information enables that.
The image in figure 5.2 depicts a plot which shows the correlation between a specific tower module ID of a tower module X(the specific tower module shown is referred to as module X) in green and the tower name in blue. It can be seen that the tower module ID of the module X uniquely defines a test tower thus enabling to distinguish the test towers. It can be seen that the test runs go from roughly 2200 up to 3000.

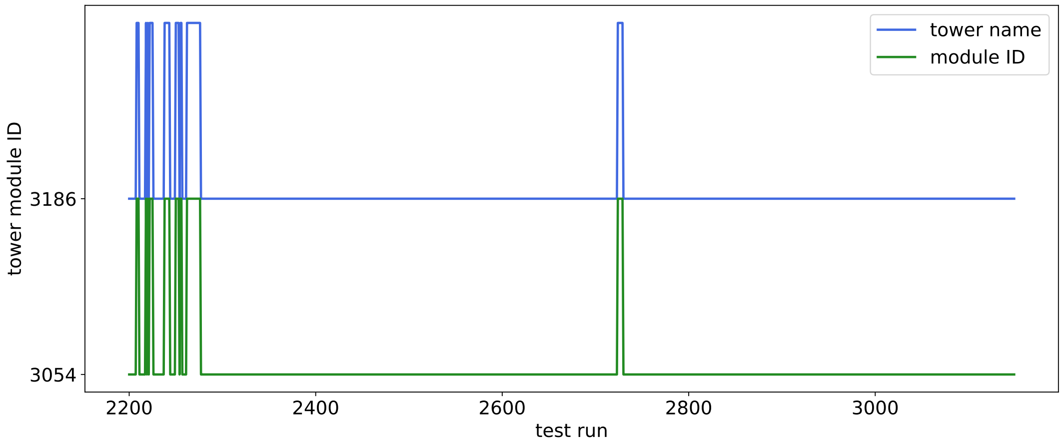


Figure 5.2: Known tower module IDs and its corresponding test tower

It shows that the tower names are defined only for the last approximately 1200 data points and missing for the previous 2200 points due to the database migration.
By knowing which test tower contains which tower module X, it is possible to fill the missing tower names. The result is illustrated in figure 5.3. For overview's sake only the first 1000 test runs are shown. With this, all data points since the commissioning of the test tower can be retrieved for a specific test tower. Without filling the tower names the dataset would contain only approximately 36% of the actual dataset.

5.6.2 Data cleaning

As a next step, some general data cleaning tasks are performed.
Since many values are returned as a string, they are converted into the **correct datatype** (i.e. a timestamp is converted from a string into a python-specific datetime datatype or a number in string format is converted into a float or an integer).
If the datasets contain any **`NULL` or `Nan` values**, they are removed by removing an

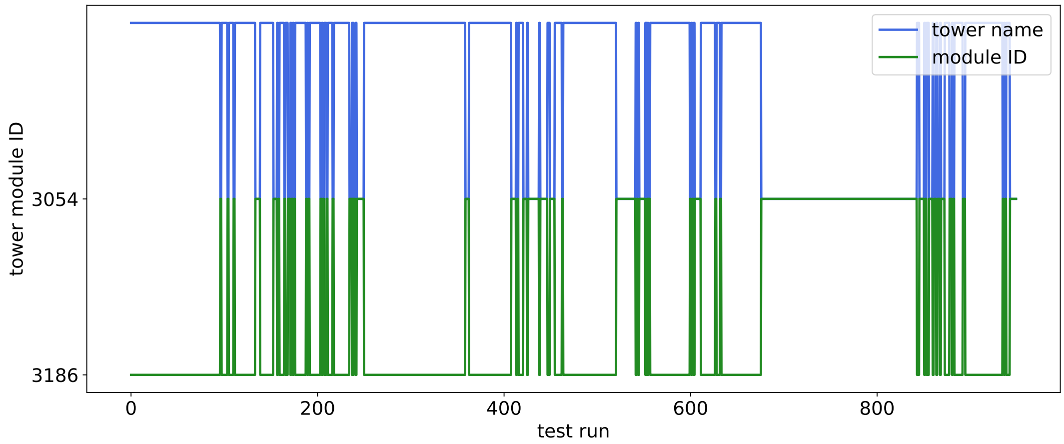


Figure 5.3: Filled tower names

entire row.
Any **outliers** in the measurement results are removed as well. An outlier is a value which lies beyond 3-times the standard deviation. This can be computed by using the zscore[34].
The data cleaning process reduces the data size by at most 30 data points which is negligible in comparison to the total size of the dataset.

5.6.3 Serial number feature

All the measurement results for subtests can be assigned to a specific DUT with a specific serial number. Using the serial number each DUT can be assigned to a production series in which it has been produced with a set number and types of components. However, there is the possibility that DUTs from different production series have slightly different measurement results due to different component types used in the assembly.
Therefore, this effect has to be removed. The idea is to calculate the mean value of the results for the first 10% of the DUTs for each production series. The reference production series is the production series of the first entry in the dataset. The effect is removed by subtracting the absolute offset of each production series mean value relative to the reference. The result of such an operation is illustrated in figure 5.4. It can be seen that there is a substantial impact of the production series dependency on the data.

5.6.4 Error feature

All the presented subtests in table 5.1 yield only one result - the absolute error of the measurement. This means that there is only one feature available for each subtest. The **absolute error** e_{abs} for each subtest measurement is given with the equation 5.1.

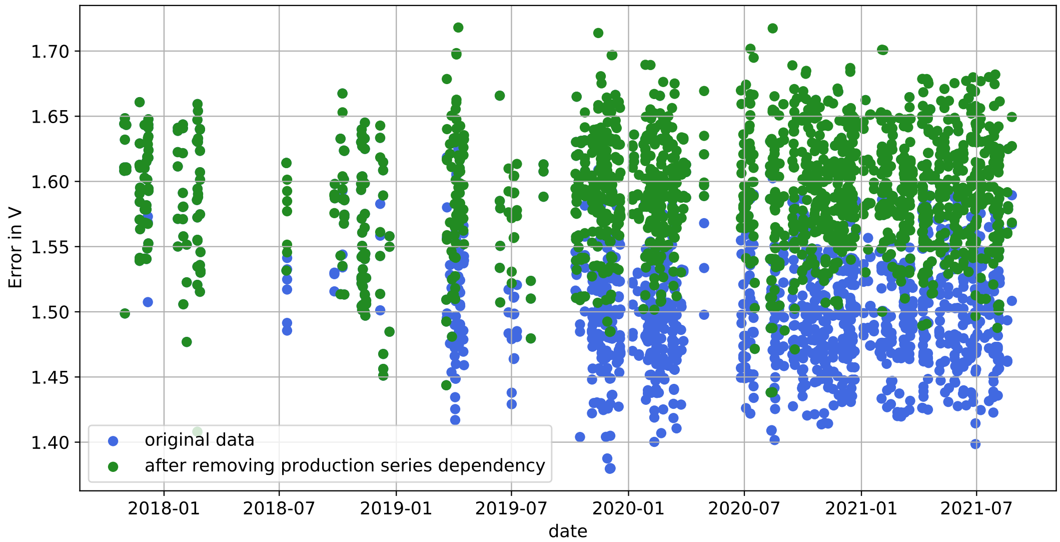


Figure 5.4: Data after removing production series dependency

$$e_{abs}(x) = x - x' \tag{5.1}$$

where:

x = Measured value of the subtest
x' = Actual or nominal value of the input

This absolute error is an unfit feature when the goal is to compare the subtest errors for different parameters and channels. Therefore, a relative feature is needed. The first approach is to use the **relative error** e_{rel} using the formula in equation in 5.2.

$$e_{rel}(x) = \frac{e_{abs}}{x'} \tag{5.2}$$

with $x' \neq 0$

This enables to compare one subtest with its counterparts with different channels and parameters.
Unfortunately, there are nominal value for some subtests that are 0. Thus, the relative value is not always an ideal choice because the base value with x' can be 0. In order to consistently use the same feature for all subtests, another type of relative value is evaluated.

The **relative percent difference** or **RPD** is still well defined even for nominal values equal 0 as seen in equation 5.3.

$$e_{RPD}(x) = 2\frac{x - x'}{|x| + |x'|} \tag{5.3}$$

with $e_{RPD} \in [-2, 2]$

The cases of subtests with $x' = 0$ are now well defined because the denominator is very unlikely to be 0. All performed calculation with the given dataset have never had the case where the measurement was exactly zero when the nominal value was also 0.
This relative value results in all relative errors to be located at the extreme values -2 or 2 if the nominal value is 0. This relative value does therefore not solve the problem entirely since the values can no longer be distinguished from one another if the nominal value is 0. A different approach is needed.

Since the goal of the thesis is partly to find trends in the time series data, a new relative value is created, called **relative error re first use**. This relative value tries to capture the factor of change of the data points relative to an initial reference point. The reference point is the value of "first use" of the tower unit test or the initial error value of the subtest at the time of its commissioning. The reference value $e_{re-first-use}(x)$ is calculated by using the average of the first 10% of the dataset and the resulting "relative error re first use"-feature is computed by using the formula in equation 5.4. The idea is illustrated in figure 5.5.

$$e_{re-first-use}(x) = \frac{x - x'}{\tilde{x}} \tag{5.4}$$

where:

$\tilde{x}$ = first use value as reference value

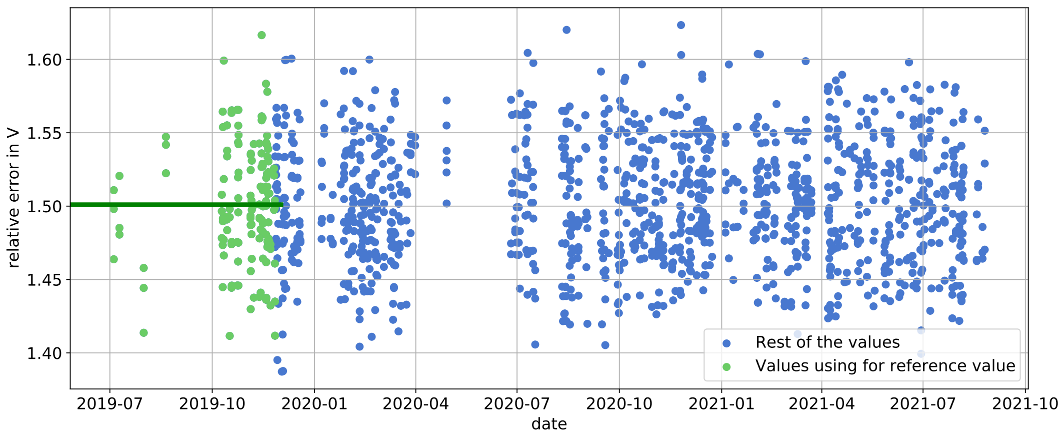


Figure 5.5: Creation of 'relative error re first use'-feature

The green points are used to calculate the mean value which results in the reference value shown as the green line. The resulting feature is plotted in figure 5.6.

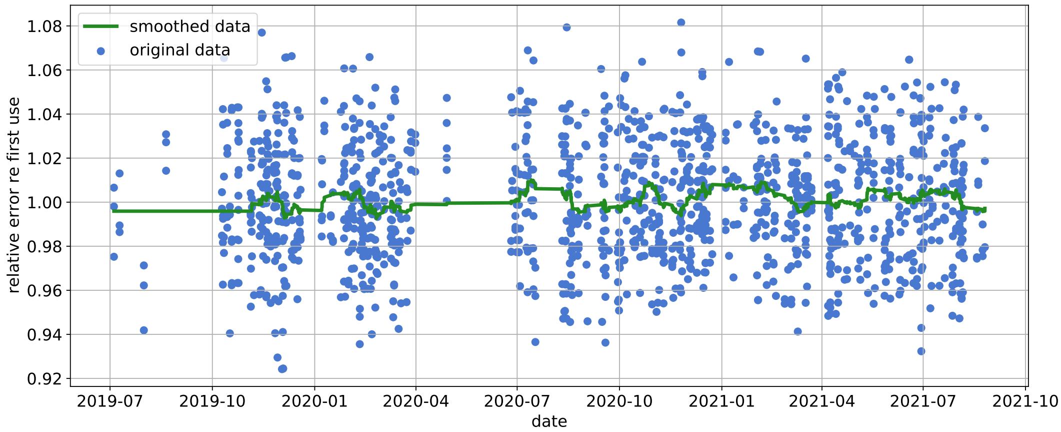


Figure 5.6: Result of 'relative error re first use'-feature

The blue points in figure 5.6 show the factor relative to the reference value, e.g. a data point with the value 1.06 means than it is 1.06 times larger than the reference value. The green line shows the result of the smoothing process for the feature.

Multiple tests and observations of this feature for the given dataset have shown that this feature is suitable to be used as a universal feature for the given datasets. A positive side effect is that it can also be used to compare different subtests with each other since it measures the factor of change relative to an initial reference value. This feature is the main feature throughout all subtests and is used to calculate other feature in the coming chapters.
One issue that was analyzed for this feature is the fact that the denominator is the reference value. The thought was that there might be a problem if the reference value is very small e.g. 1×10^{-9}. For such a small value the result would be enormous and since such mean values occur quite often, this might result in a problem. The observations have shown that this is not case because the term in the numerator proportionally decreases/increases or to be more exact, it lies in the same value range as the reference value. Consequently, if the reference value is very small, the numerator is proportionally equally small, thus negating the issue.

It can be seen in the figure 5.5 that there is a lot of noise on the signal. All the analysis techniques and algorithms that are used in the later chapters are sensitive to noise. Thus, a filtering/smoothing is performed on the datasets. Since all subtests have a similar noise-problem, the filtering is applied to all subtests.

5.6.5 Timestamp feature

The datasets have also timestamps as a feature. Those timestamps show accurately up to seconds when the subtest was started. In the context of this research such accuracy is not relevant since it is assumed that the trends are rather long-term changes. Any

changes that are short-term are presumed to have a different source of error.
Another inconsistency in the dataset due to the timestamps is the arbitrary sampling time. In order to counteract this, all measurement results within one day are summed up into one value as an average. This has a filtering effect since many data points are now mean values. This filtering effect is shown in figure 5.7. The effect can be better seen in the distribution plot in 5.8 where a decrease in standard deviation is noticeable with a 15% decrease.

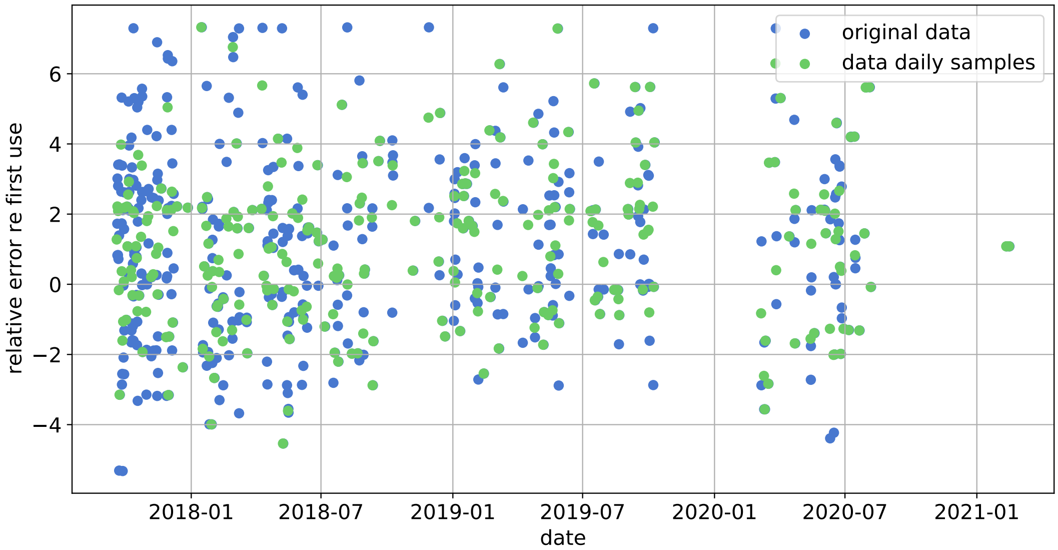


Figure 5.7: Filtering effect after summing up into daily samples

5.6.6 Moving average

It can be seen in figure 5.7 that the dataset still has many short-term variations, therefore an additional filter is applied using **moving averages**. The moving average for a dataset can be computed in different ways, two of which are considered in this research.

Rolling moving average

The rolling moving average, also known as *Simple Moving Average*, is the unweighted mean of the previous N date points. By selecting a desired sliding window size N the smoothing effect can be adjusted [23]. The selected window is rolled over the data set and each sequence of values inside this window is used to compute one rolling average value for the time t using the formula in 5.5.

$$SMA_t = \frac{x_t + x_{t-1} + x_{t-2} + ... + x_{N-(t-1)}}{N} \tag{5.5}$$

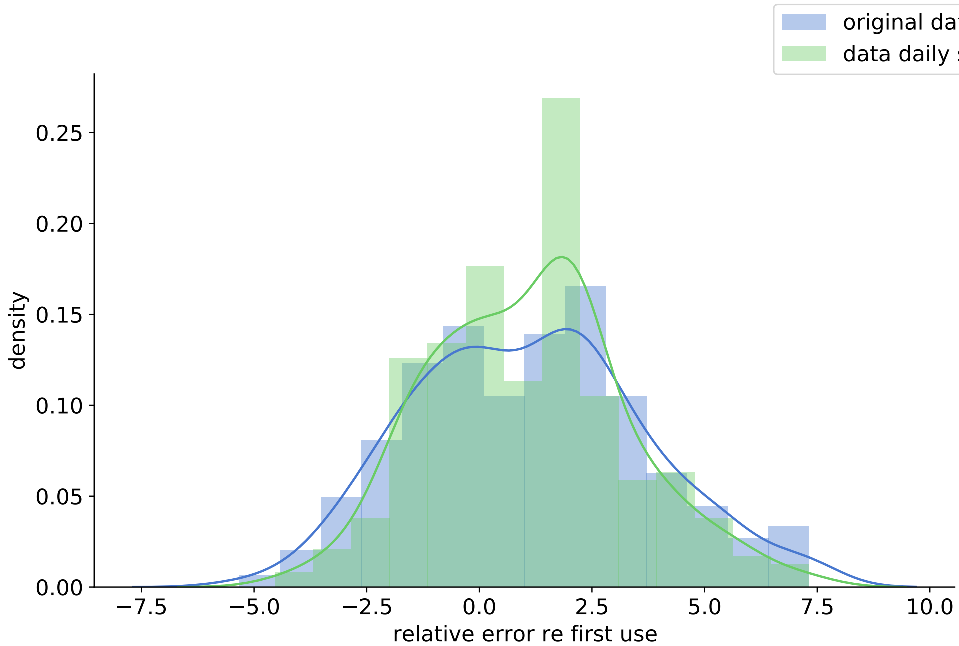


Figure 5.8: Distribution plot for filtering effect after summing up into daily samples

All data points for $t < N$ needed to be handled differently since the window size is too large for those points. Those data points can either be left unfilled or a cumulative moving average can be performed for those points.

Cumulative moving average

This type of moving average starts with the first value and uses the ever-expanding dataset to calculate the mean value. The Cumulative Moving Average (CMA) is the unweighted mean of the previous values up to the current time t. As opposed to the constant window size in the SMA, the CMA has a ever-growing window size as the time passes [23]. The computation can be done using the equation 5.6.

$$CMA_t = \frac{x_1 + x_2 + x_3 + ... + x_t}{t} \tag{5.6}$$

Since the CMA takes all previous data points into account when calculating the average, it is not an ideal option for analyzing trends.

Therefore, for filtering and smoothing of the subtest data points, the rolling moving average (SMA) is used.
The results of the filtering can be seen in 5.9. The distribution plot in figure 5.10 paints also a clear picture showing that the standard deviation of the smoothed signal has greatly decreased compared to the original data indicating a successful filtering. The filtering achieved a reduction of approximately 80% in the standard deviation.

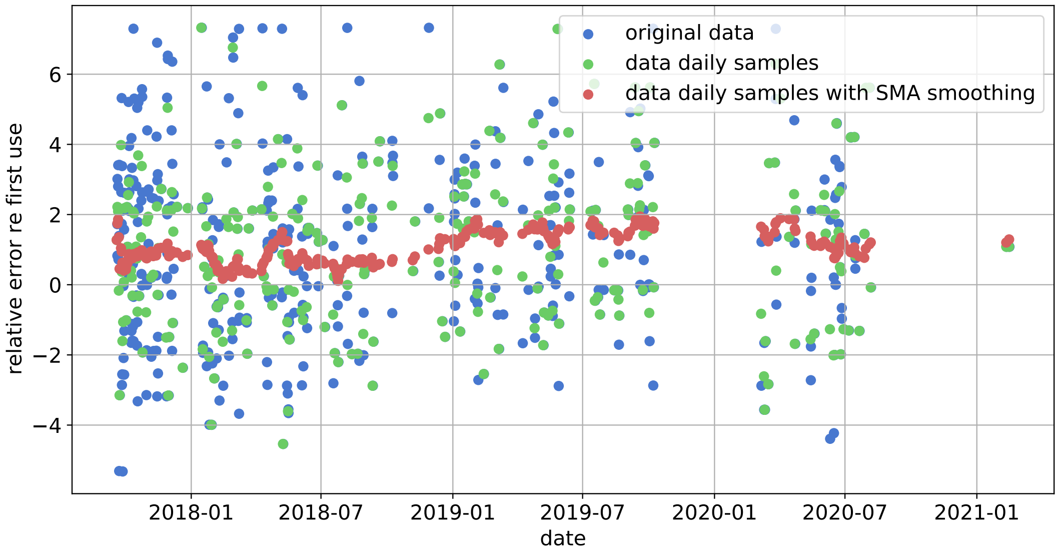


Figure 5.9: Filtering effect after SMA smoothing

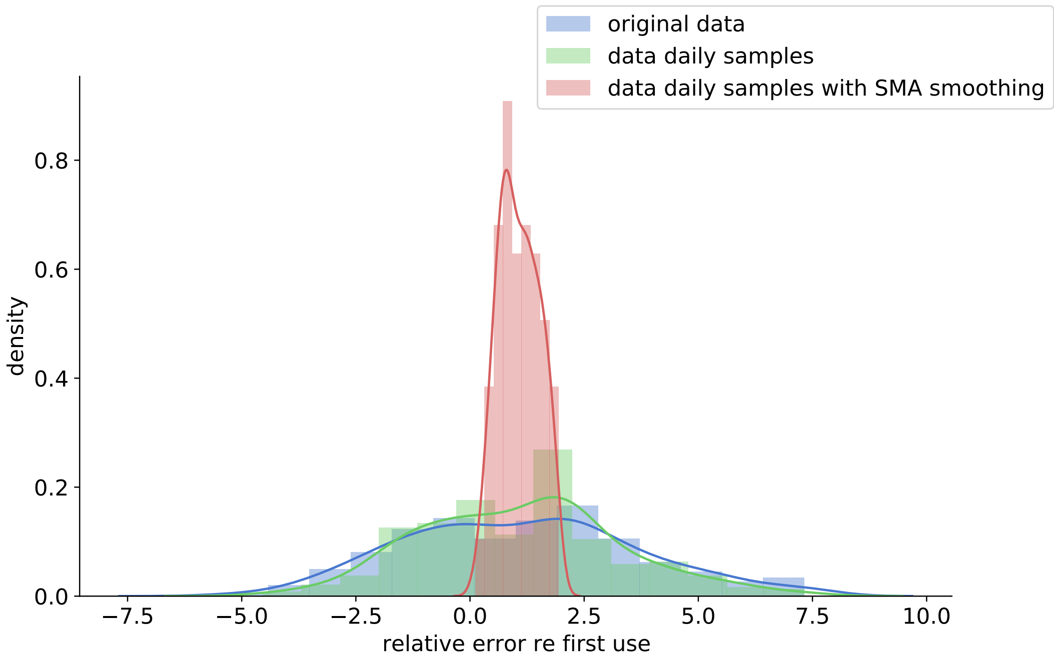


Figure 5.10: Distribution plot for filtering effect after SMA smoothing

Since the "relative error re first use"-feature has been designated to be the main feature, it can be attempted to derive additional features from it.
For this research the following classes of features are considered:

(1) Time domain features

(2) Frequency domain features

(3) Time-frequency features

5.6.7 Time domain features

Table 5.4 shows the time domain features that have been computed for the subtests. The input `x` for the computations of the feature are the samples within each day, similar to the chapter 5.6.5.
Time domain features in general have a good interpretability and are useful when analyzing trends in datasets.

Feature	**Computation**
Mean	`mean(x)`
Standard deviation	`std(x)`
Skewness	`skew(x)`
Kurtosis	`kurtosis(x)`
RMS	`rms(x)`
CrestFactor	`max(x)/rms(x)`
ShapeFactor	`rms(x)/mean(abs(x))`
ImpulseFactor	`max(x)/mean(abs))`
MarginFactor	$\texttt{max(x)/mean(abs))}^2$
Energy	$\texttt{sum(x}^2\texttt{)}$

Table 5.4: Time domain features

Each feature is smoothed using the previously mentioned rolling average method with SMA. This ensures that the noise is removed and only the trend remains.

5.6.8 Frequency domain features

In order to get insight into the frequency domain of a feature the **Fourier Transform** is used [22], or the computationally cheaper version - FFT. The analysis of these features is performed in chapter 6.1.

5.6.9 Time-frequency features

For the time-frequency analysis the **Short Term Fourier Transform**[36][22] and the **Wavelet Transform**[44][22] are used. These analyzes enable to inspect the time and frequency domain simultaneously.
The evaluation of the effectiveness or usability of this feature relies on the visual interpretation of the STFT and the Wavelet Transform plots, also referred to as *spectrograms*.

5.7 Theoretical background - comparison techniques for features and signals

The current dataset consists of multiple parameters, channels, functionalities that are tested in each subtest etc. resulting in a total number of subtests of 259. Combined with the number of newly introduced features in the previous chapter the data mining process becomes less manageable. Moreover, initial observations have shown that the subtest features partially contain redundant and also low-quality data(i.e. the trend behaviour of the data is weak). The goal is to create a dataset which does not contain subtests that a redundant or subtests which do not contribute any information to the dataset due to their low data quality. This goal can be achieved by performing an abstraction of the dataset.
Specific techniques are used in order to perform an abstraction, reduce the number of subtests, remove redundances and get rid of low-quality subtests. Each of them yields a numerical value which is quantifiable and comparable thus being a good metric.

5.7.1 Dynamic time warping

As a measure for similarity of two time series signals the normalized time distance D is used [37] which can be calculated after computing the **Dynamic Time Warping**[21].

The Dynamic Time Warping, also referred to as DTW, is a popular algorithm due to its extreme efficiency as a time-series similarity measure. Generally known distance functions try to align the i-th point of one time series with the i-th point of another time series which in general produces a poor similarity score. DTW, however, uses "elastic" transformation of time series which minimizes the effects of shifting and distortion in time, thus detecting similar shapes with different phases [35]. The image in 5.11 gives a rough comparison between a euclidean matching and the DTW matching. Two time series signals are considered similar if the normalized time distance D is smaller than 0.1.

5.7.2 Monotonicity

When trying to implement a PdM it is important to know whether a feature is suitable for prognostic purposes or as a condition indicator.
The monotonicity is such a metric. It tries to evaluate the trend of a feature, with the assumption that the system cannot self-heal [20][31]. It is given as the absolute difference between positive and negative derivatives of the feature as shown in the equation in 5.7.

$$M = \left| \frac{\text{Number of } \frac{\Delta x}{\Delta t} > 0}{n-1} - \frac{\text{Number of } \frac{\Delta x}{\Delta t} < 0}{n-1} \right|, M \in [0,1] \tag{5.7}$$

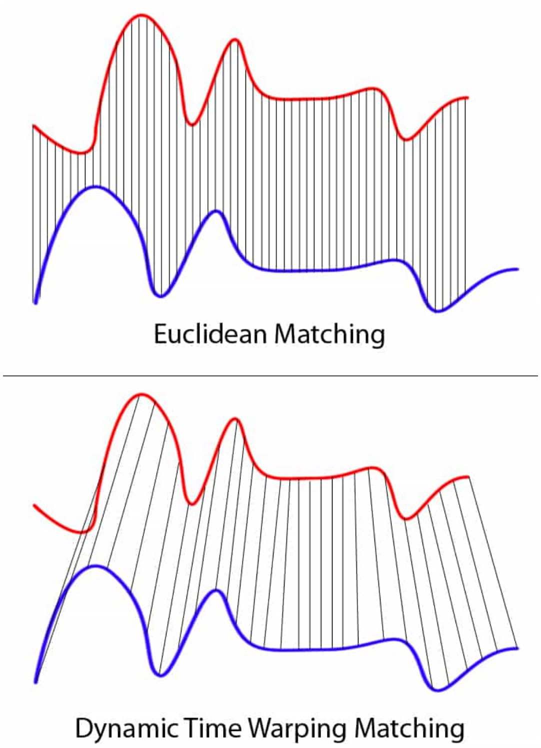


Figure 5.11: Euclidean Matching vs. DTW Matching[47]

where $\frac{\Delta x}{\Delta t}$ represents the derivative of the feature or the be more exact the difference quotient of the feature and n is the size of the dataset. The higher the value of M, the higher the suitability a feature[4] for prognostics.
The calculation in python is performed using the source code shown in the appendix in listing 9.3.

5.7.3 Trendability

Another method to evaluate the suitability of a feature for PdM is the trendability which is computed using the equation 5.8.

$$R = \frac{n(\sum_{i=1}^{n} x_i t_i) - (\sum_{i=1}^{n} x_i)(\sum_{i=1}^{n} t_i)}{\sqrt{[n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2][n\sum_{i=1}^{n} t_i^2 - (\sum_{i=1}^{n} t_i)^2]}}, R \in [-1, 1] \tag{5.8}$$

The trendability is related to time and quantifies the correlation of the degradation trend with the operation time of the system [20].
A high absolute value of R indicates a high correlation of the feature with the operation time which in turn corresponds to a higher trend. A high negative R means a negative correlation between the feature and time but indicates a trend nonetheless. In this

research mainly the absolute value for trendability is inspected since they sign of the correlation is irrelevant which is given in equation 5.9.

$$R = \left| \frac{n(\sum_{i=1}^{n} x_i t_i) - (\sum_{i=1}^{n} x_i)(\sum_{i=1}^{n} t_i)}{\sqrt{[n \sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2][n \sum_{i=1}^{n} t_i^2 - (\sum_{i=1}^{n} t_i)^2]}} \right|, R \in [0,1] \tag{5.9}$$

The calculation in python is performed using the source code shown in the appendix in listing 9.2.

5.7.4 Thresholds for trend evaluations

For the monotonicity there is a documented value which can be used as a threshold (i.e. feature with $M > 0.3$ is considered to have good prognostic utility) when evaluating features for it [45] whereas for the trendability metric no definite value could be found. In order to have a rough idea what values a typical PdM system has, the following sections describes two PdM projects including the evaluation of their features for trendability.

Reference projects

The selected reference projects are the **Motor Degradation** and the **Wind Turbine Degradation** project. The datasets for the reference projects can found in Turbofan Engine Degradation Simulation Data Set[28] and in Wind Turbine High-Speed Bearing Prognosis from Mathworks[45].

The **Motor Degradation** project has multiple run-to-failure datasets which can be used to train a ML model. The dataset has 21 features, 14 of which have good predictive utility. The results for the monotonicity and trendability analysis for those features can be seen in figure 5.12. Figure 5.13 shows the plots for some of the parameters with the original data points and the smoothed version. It can clearly be seen that there is obvious trend behaviour in the data thus it can be assumed that if both metrics show values above their respective thresholds, it is more likely that the trend behaviour is existent.

The **Wind Turbine Degradation** project uses multiple condition monitoring samples of the system to create a PdM. The samples are time series data points which are used to derive multiple features as explained in more detail in [45]. The monotonicity and trendability values for this project are shown in figure 5.14.
The selected final features for PdM in this project were `Mean`, `ShapeFactor`, `MarginFactor` and `Kurtosis`. Figure 5.15 shows that the selected features `Mean` and `Kurtosis` visibly show a trend behaviour whereas the feature `Skew` does not have a very pronounced trend behaviour which is can also be seen by the low trendability and monotonicity value.

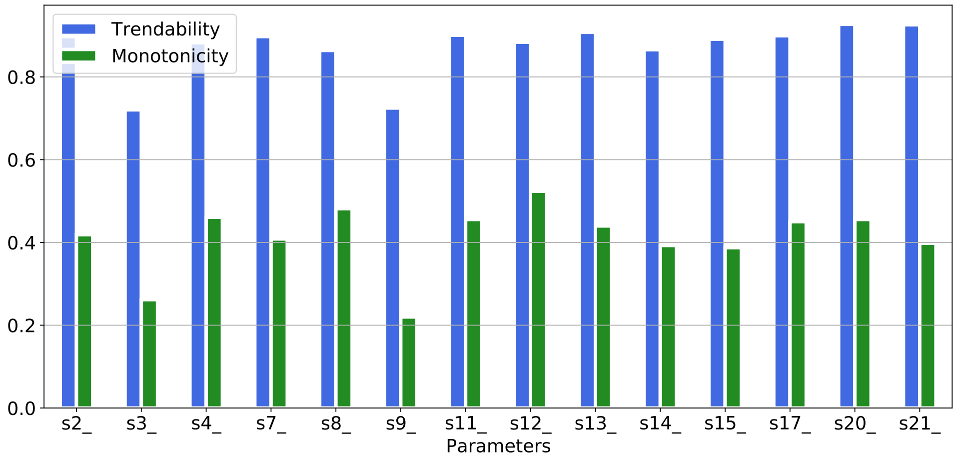


Figure 5.12: Trendability and monotonicity results for motor degradation

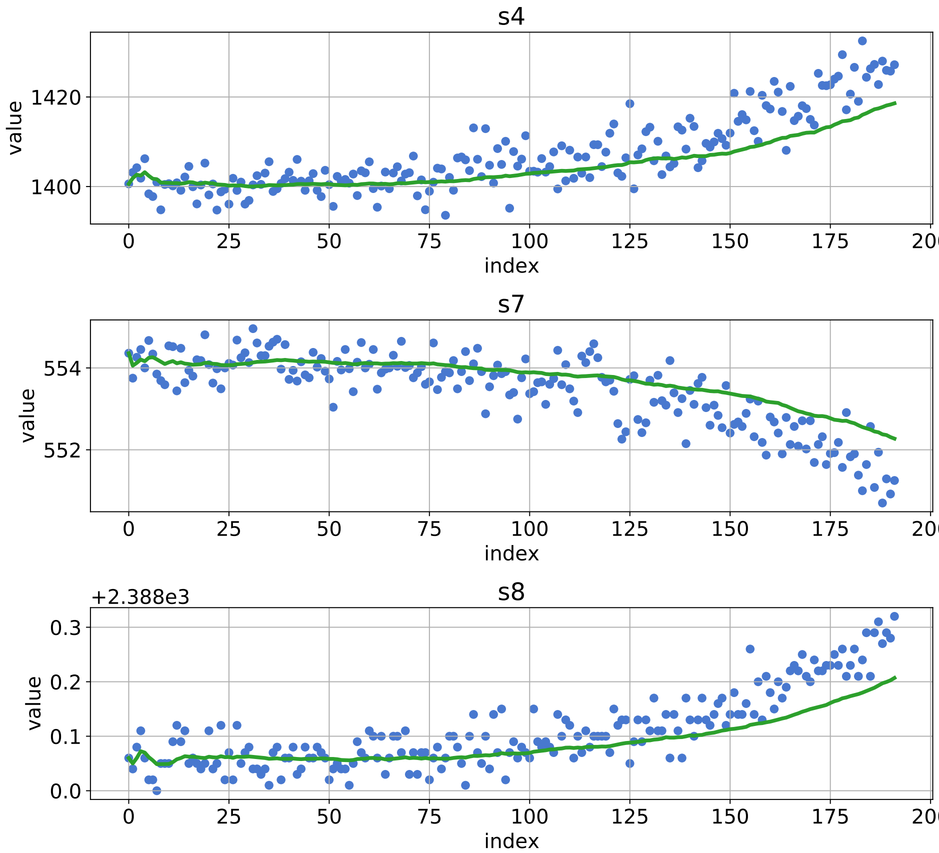


Figure 5.13: Plots for the s4, s7 and s8 parameters of the motor

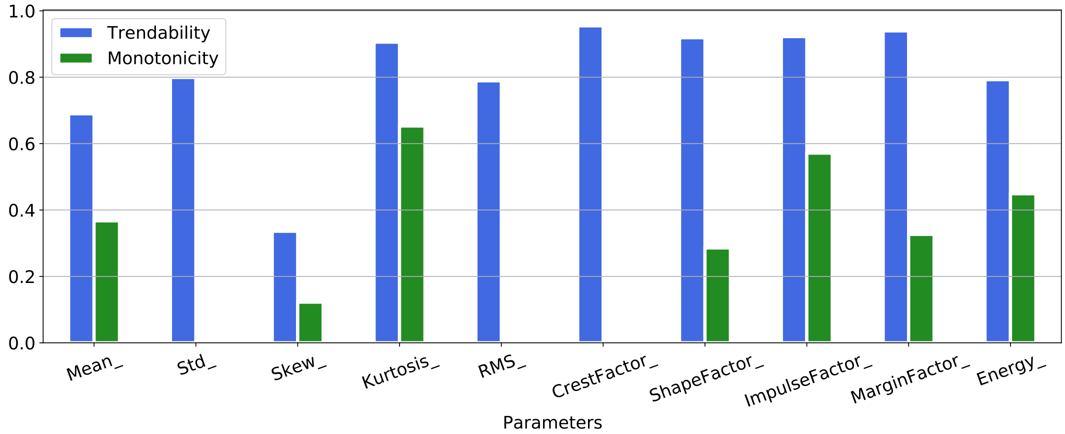


Figure 5.14: Trendability and monotonicity results for wind turbine degradation

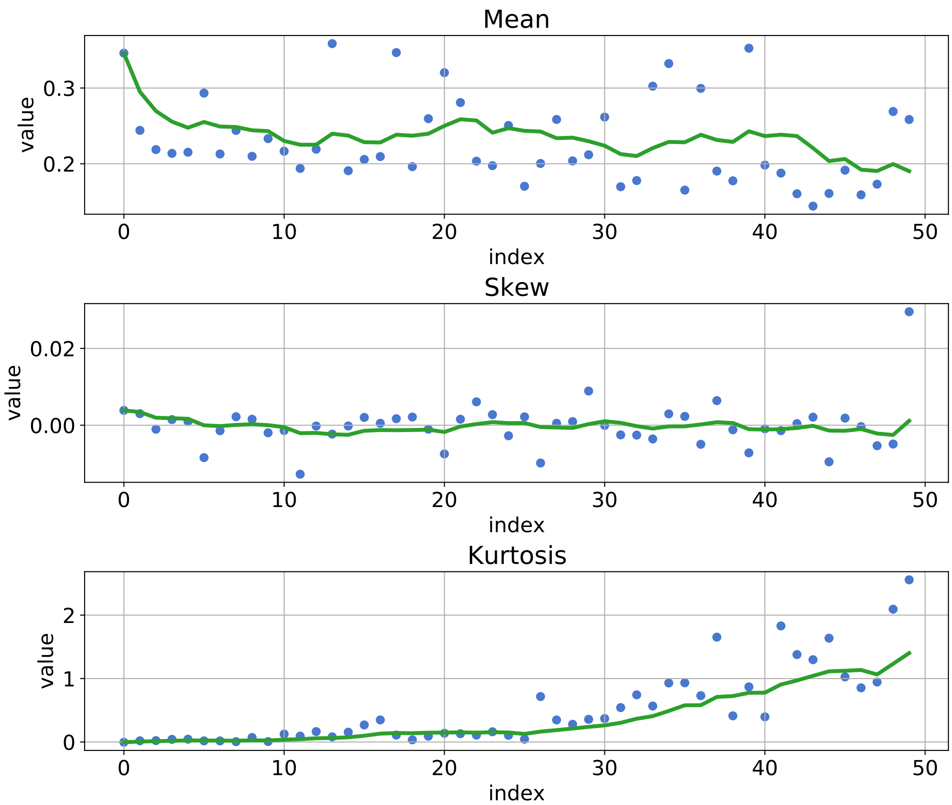


Figure 5.15: Plots for the `Mean`, `Skew` and `Kurtosis` parameters of the wind turbine

Considering the results of the reference projects, the threshold for the trendability metric is chosen to be at 0.5, meaning a good prognostic value for the trendability is for $R > 0.5$.

5.8 Dimensionality reduction and abstraction of the unit test

In table 5.1 it can be seen that the number of tested functionalities is not very large, just 15. The problem is that each functionality has multiple parameters and multiple channels. These are dependencies which, if removed, can reduce the dimensionality. All the dependencies that are analyzed are listed below.
It should be noted that the goal is to find features in datasets that have a certain trend which could indicate a degradation of the test tower as mentioned in chapter 2. With this in mind, the subtests can be separated into classes or summed up by removing the following characteristics.

- **Channel dependency:** If the relative trend of the measurement results is similar for each channel, the channels can be summed up into one channel. This moves the redundant subtests by combining their data into one virtual channel.
- **Parameter dependency:** If the relative trend for each parameter is similar, the parameters can be summed up in to one virtual parameter. If the parameter show different trend behaviours, the parameters with the best trend is selected while the others are removed. For evaluation of the trends the monotonicity and trendability metrics can be used.
- **Miscellaneous reasons:** Other reasons for removing subtests can be the inability to properly access its data in the database or the general uselessness of its data(i.e. the data quality is low) in the context of this thesis.

In order to reduce the dimensionality of the feature space a feature ranking can be performed. This will yield a numerical importance metric for each feature allowing to remove less relevant features. By following up this process with an additional dimensionality reduction technique, such as PCA[3][21], the feature space can be optimized even further.

5.8.1 Abstraction of the unit test

Channel dependency

Several subtests as seen in table 5.1 are used on multiple different channels. If the results are similar for all the channels, they can be summed up into one “virtual” channel. This does not reduce the number of data points but it reduces the number of subtests. For measuring the similarity of the channels, the **Dynamic Time Warping** technique is applied as explained in chapter 5.7.1.
The comparison of the different channels is performed with the following assumptions:

- A `MinMaxScaler`[39] is applied if the scaling of the signals is visibly large. This removes any scaling problems for the DTW.
- The reference time series signal is always the first signal.
- The other channels are compared with the reference and the D value is calculated
- The test tower ETT1 is used as the representative for both towers

An example of a DTW analysis for a subtest can be seen in figure 5.16. The DTW matching has a normalized time distance D of 0.011.

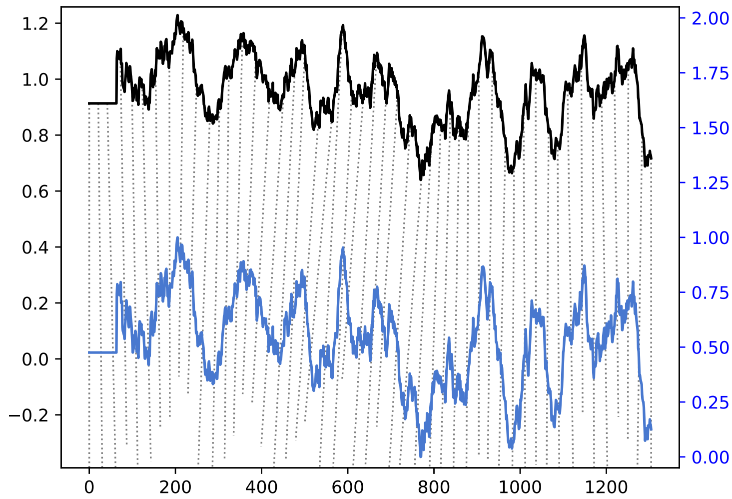


Figure 5.16: DTW matching for M2Sub2 parameter B

The table in 5.5 shows the results for the normalized time distance D with $D \in [0, 1]$. The similarity is large for small D values and vice versa. The threshold for a good similarity is selected to be $D < 0.1$.

The table 5.5 gives a good overview that all subtests with multiple channels have very similar signals and can therefore be summed up into one channel. It gives the mean value for the similarity for all channels and all parameters in one subtest. Additionally, the max value was depicted as well in order to know the span of the values. It can be observed that the similarity mean values for all subtest are well below 0.1 and none of the maximal values exceed the threshold.
Since all channels give similarly looking and thus similarly trending signals, the channels are combined into one channel.
The only exception are the subtests from the M1 module. For that module, there is a distinct peculiarity. The channels from 1-3 have a higher accuracy than the channels 4-6 which can be found in the datasheet[10]. Therefore, the six channels of the M1 are

Module	Subtest	Mean of D	Max of D
M1	Sub1	0,004386432	0,00689241
	Sub2	0,002524475	0,004528368
	Sub3	0,002563445	0,004083599
M2	Sub1	0,000214568	0,032679692
	Sub2	0,000130241	0,019919284
	Sub3	8,7845E-05	0,013587447
	Sub4	1,72072E-05	0,002599619
	Sub5	0,000210803	0,032182511
	Sub6	0,00012282	0,018863621
	Sub7	2,92401E-05	0,004455421
	Global mean value:	0,001028708	0,013979197

Table 5.5: Overview of time distances D for each subtest with multiple channels

summed up into two channels.
In order to select one of those two virtual channels, a trendability test is performed as described in the chapter 5.7.3. Since the analysis of trendability is one of the goals for this research, the class of channels with better trendability value is selected. The focus is on the trendability of the "relative error re first use"-feature since it is comparatively easier to interpret than other time domain features.

The figure in 5.17 shows the difference in the trendability for the different channel classes for different parameters. The monotonicity values can also be seen on the image but since they are so small, they are not considered.
It can be observed that the channel class "Ch1-3" has a smaller or worse trendability metric compared to the "Ch4-6" class for most of the parameters. This relationship is true for small as well as for larger parameter values.

Conclusively, the channel class "Ch4-6" are selected since they show more trendability traits and because the trendability values are higher. This means that the result of the channels 4-6 can be considered to be from the same channel.

Since we can remove all channel dependencies from the test tower dataset, the resulting dataset has the following structure as seen in table 5.6. The number of subtests has now been reduced from 259 to 100 subtests without changing the actual size of the dataset.
The next step is to look at the parameters and deduce whether all of them are unique or if they can be summed up into classes as well.

Parameter dependency

In order to remove the parameter dependency, the trendability metrics are evaluated for each subtest. The classification into subtest classes depends on the similarity

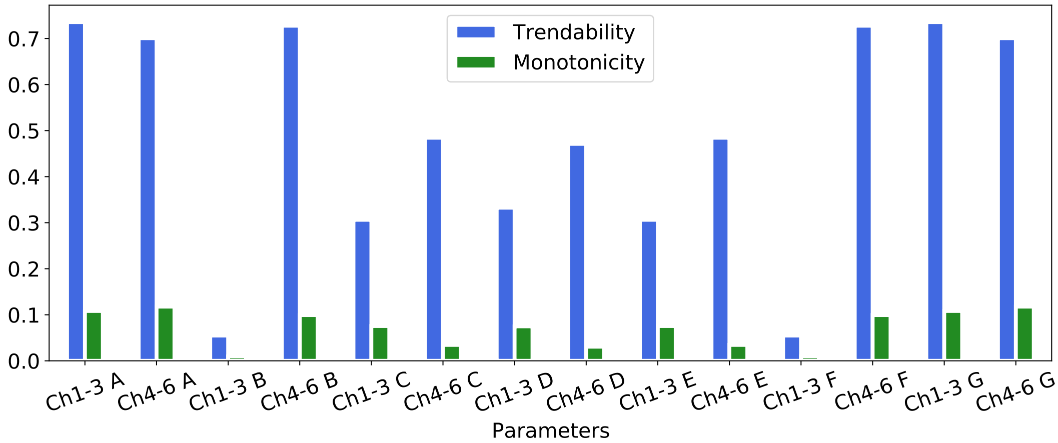


Figure 5.17: Trendability for the different channel classes

Module	Subtest	Number of Parameters	Channels	Types of Result Units	Sum of Subtest Runs
M1	Sub1	7	1	1	7
	Sub2	1	1	2	2
	Sub3	1	1	2	2
M2	Sub1	8	1	1	8
	Sub2	8	1	1	8
	Sub3	6	1	2	12
	Sub4	4	1	2	8
	Sub5	3	1	2	6
	Sub6	4	1	2	8
	Sub7	1	1	2	2
M3	Sub1	8	1	1	8
	Sub2	4	1	1	4
M4	Sub1	12	1	1	12
	Sub2	12	1	1	12
	Sub3	1	1	1	1
				Sum of subtests:	100

Table 5.6: Overview of subtests after removing channel dependency

metrics results of DTW. But the signals for different parameters do not show a strong similarity as it was the case with the channels. This dissimilarity can be seen in the coming sections in the respective plots.
Furthermore, the signals show no to little trend behaviour, therefore, only the signal with the best trendability and monotonicity metrics are selected as the “representative parameter”. Of course, by removing parameters some datasets are ignored and the remaining data size decreases. But the assumption is that due to the small trendability and monotonicity values those discarded parameters are rather noise than useful information.

The subtests for the modules M3 and M4 are not analyzed in this chapter even though they do contain multiple parameters for each subtest. There is a different reason to drop them which is discussed in chapter 5.8.1.

The abstraction happens by selecting the best trendable parameter for each subtest using the trendability analysis on the "relative error re first use"-feature.
Here are the results for each subtest.

M1 Sub1

The figure 5.18 shows the signal paths for the different parameters. Visually, it cannot be determined which one has the best trend behaviour.

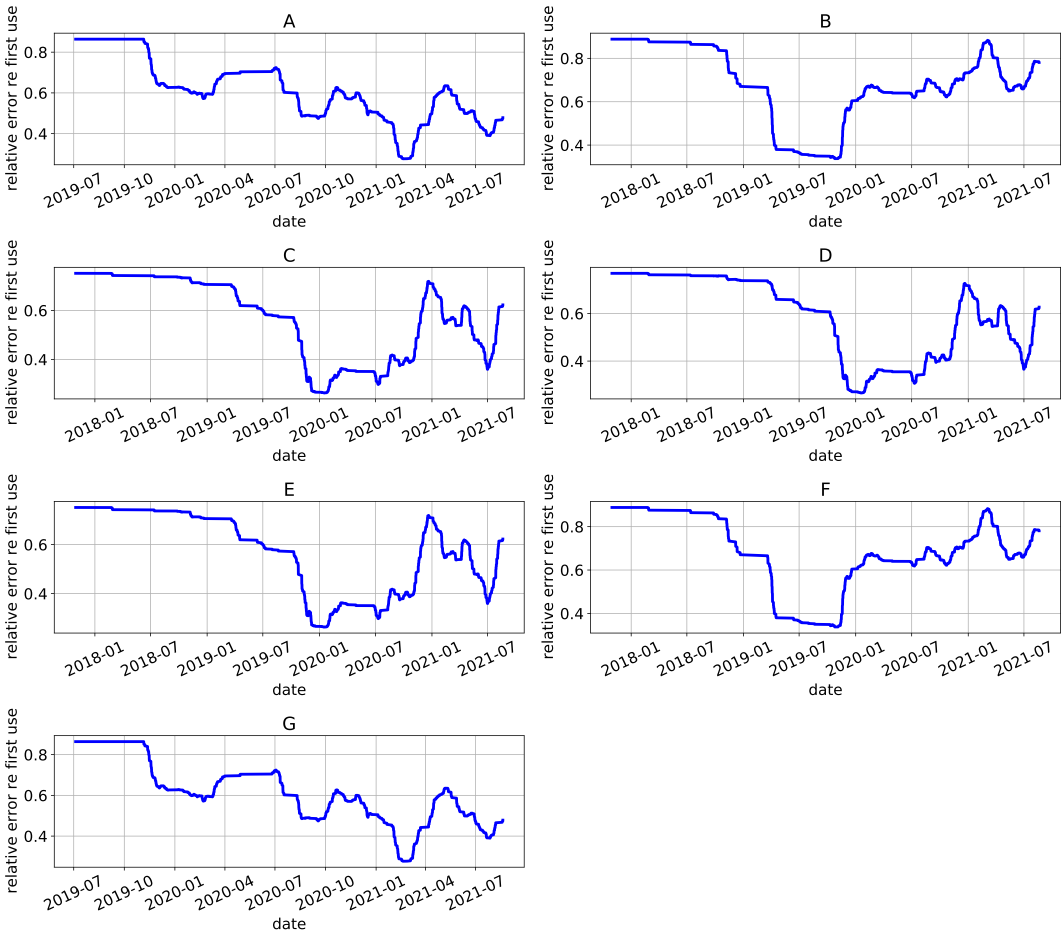


Figure 5.18: M1 Sub1 signals

The image 5.19 depict the results for the monotonicity and trendability analysis respectively. Here it is easier to determine the parameter with the best trend behaviour.

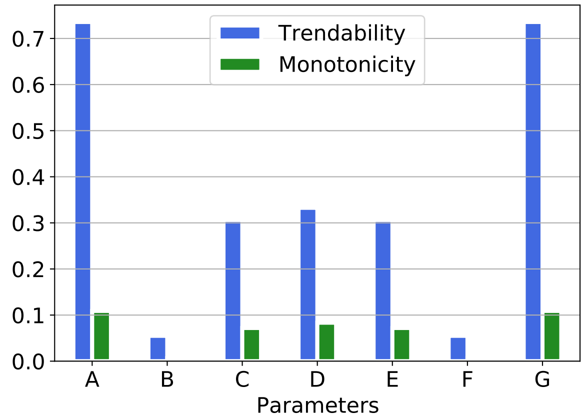


Figure 5.19: M1 Sub1 trendability and monotonicity analysis

It can be seen that the parameters with higher values do not necessarily show a more obvious trend. The **D parameter** has the highest value and therefore is chosen as a representative signal for the entire M1 Sub1 subtest.
By taking the thresholds for the trendability and monotonicity into account which were discussed in the sections 5.7.2 and 5.7.3 it can be seen that none of the parameters pass. But in this stage simply the best parameter with the high metric values is chosen without considering the thresholds.

M2 Sub1

The signals are depicted in figure 5.21. For these signals, the D parameter actually has some kind of trend behaviour which can even be detected visually.

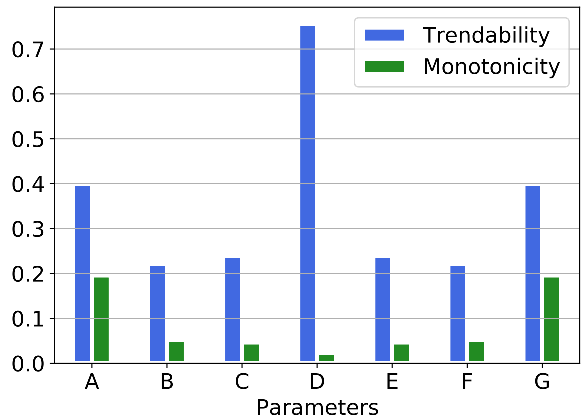


Figure 5.20: M2 Sub1 trendability and monotonicity analysis

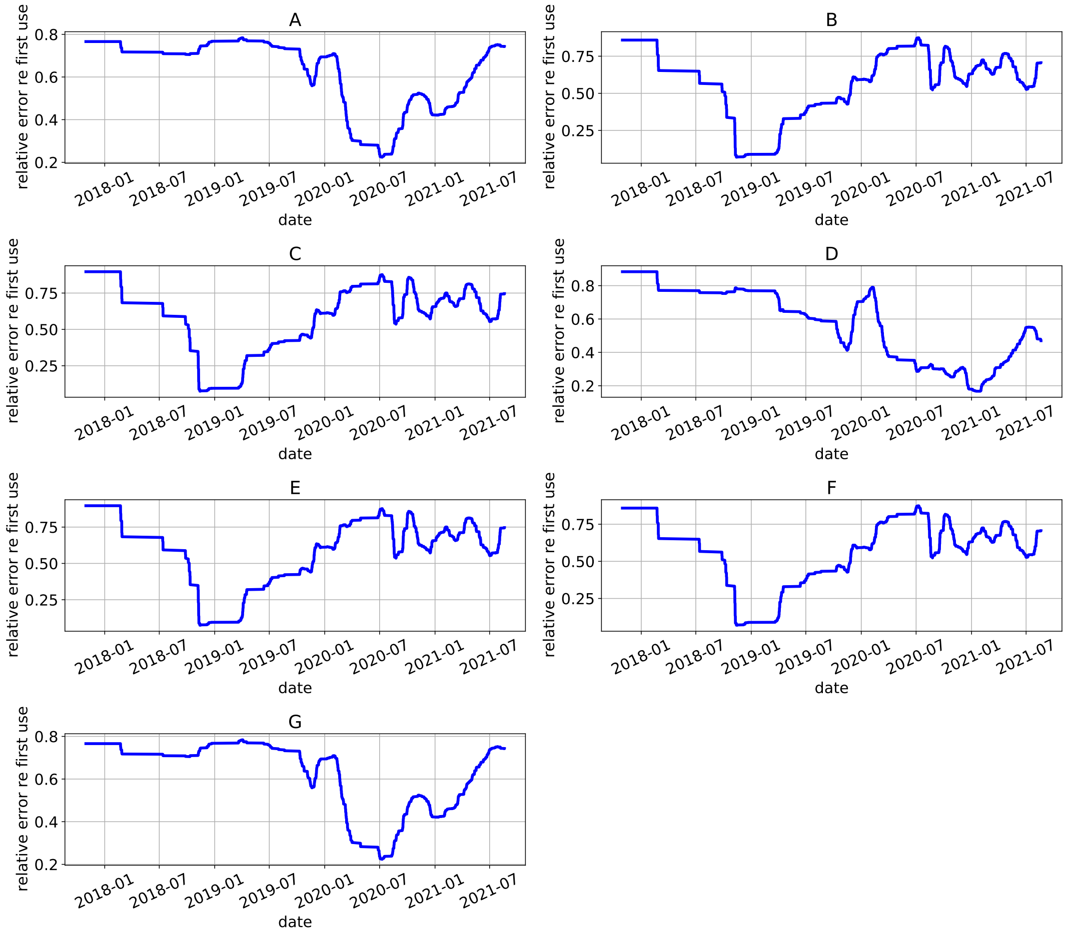


Figure 5.21: M2 Sub1 signals

It can be observed in figure 5.20 that the A and G parameter have the best monotonicity values but, the **D parameter** shows better results for the trendability analysis. Visually, the A parameter seems to have a more obvious trend since 07-2020 whereas D shows a trend over the entire timeline. Since the focus is the entire timeline, the D parameter is selected.

M2 Sub2

This subtest has similar signal trends to the previous subtest as seen in figure 5.22. The analysis results are shown in figure 5.23.
Even the trendability analysis results are quite similar, therefore the parameter with the highest value, the **D parameter** is chosen. Visually, D shows the most obvious trend compared to the other parameters. The monotonicity values are the highest for the B and F parameters, but the plots show a less pronounced trend behaviour compared to D.

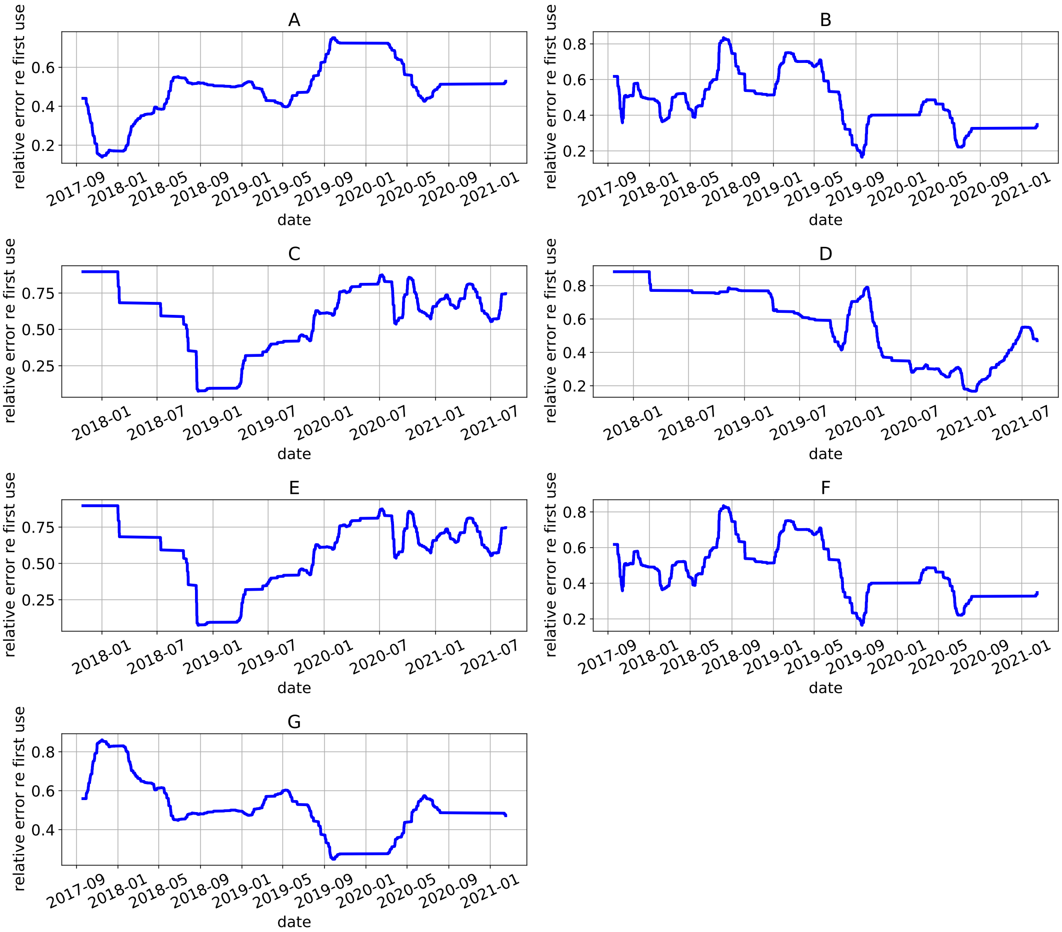


Figure 5.22: M2 Sub2 signals

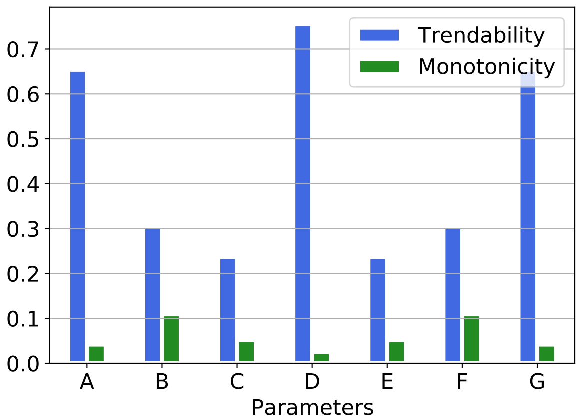


Figure 5.23: M2 Sub2 trendability and monotonicity analysis

M2 Sub3

It can be observed that the all parameters show a trend behaviour as seen in figure5.24 which is also backed up by the trendability and monotonicity metrics seen in the bar plots in figure 5.25. The highest value for both the monotonicity and the trendability can be attributed to the **parameter A**, therefore, it is the chosen representative. Interestingly, all the trendability value are within the trendability threshold of 0.5 and even the monotonicity values almost reach the monotonicity threshold of 0.3.

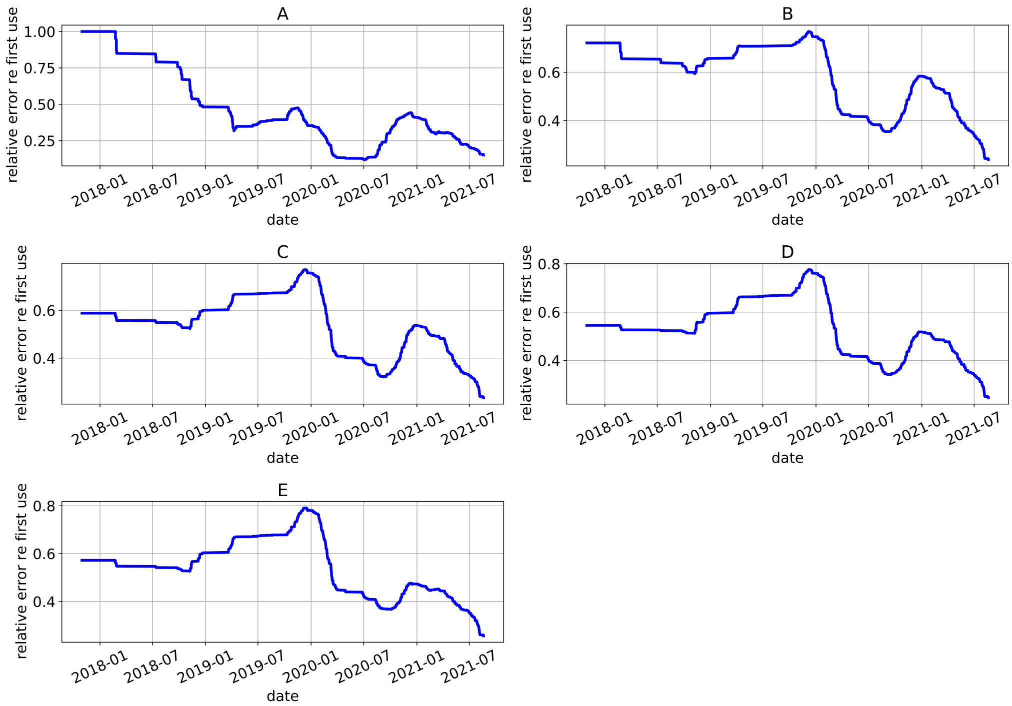


Figure 5.24: M2 Sub3 signals

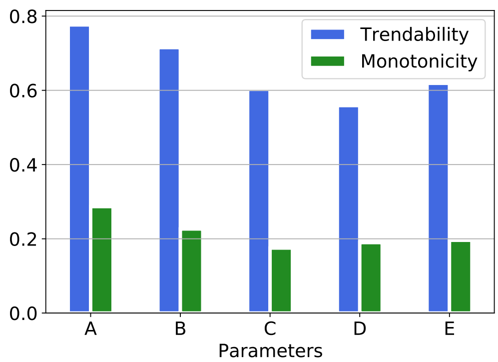


Figure 5.25: M2 Sub3 trendability and monotonicity analysis

M2 Sub4

The figure in 5.26 illustrate the signals. The results for the analyzes can be seen in figures 5.27.
Visually, no evident trend behaviour can be detected in the plots which is confirmed by the low monotonicity values. Since the **D parameter** has a relatively high value and is above the trendability threshold, it is selected as the representative parameter.

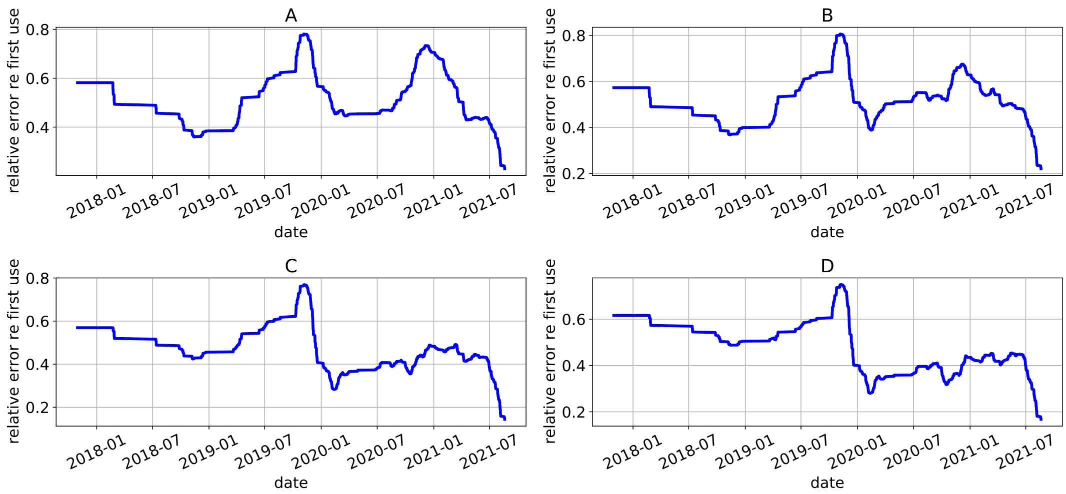


Figure 5.26: M2 Sub4 signals

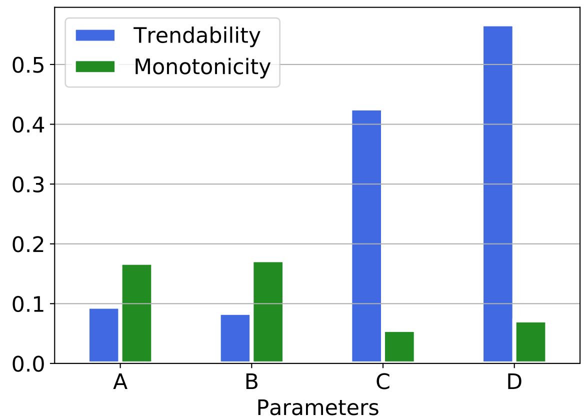


Figure 5.27: M2 Sub4 trendability and monotonicity analysis

M2 Sub5

As for the previous subtest, this subtest does not show any pronounced trend behaviours in the plots shown in 5.28. When observing the trendability and monotonicity analysis results, it can be seen that C has the best overall results. But it should be noted that the C parameter is a fairly new parameter which was introduced only in around 02-2021. Therefore, this parameter cannot be used as a representative for the entire timeline. The remaining best parameter is the **A parameter** which is selected.

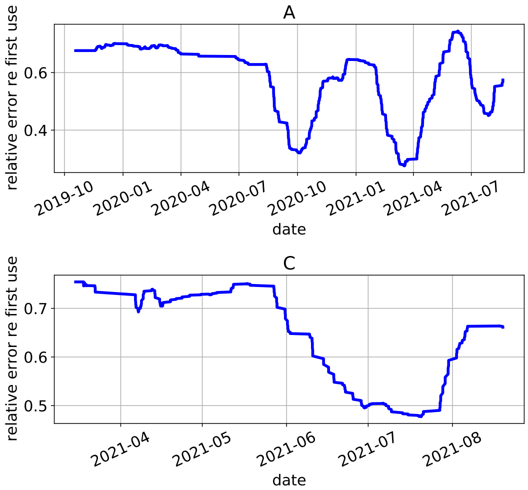


Figure 5.28: M2 Sub5 signals

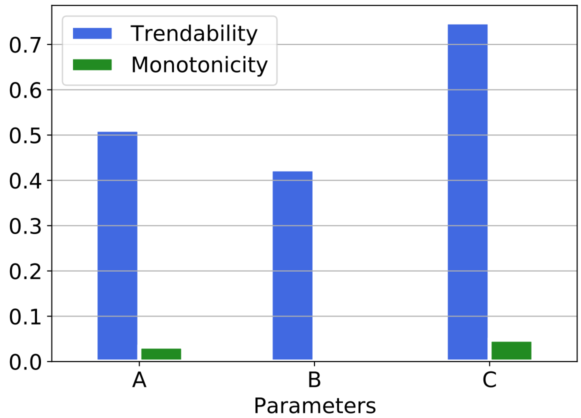


Figure 5.29: M2 Sub5 trendability and monotonicity analysis

M2 Sub6

Similarly, to the previous subtest, this subtest contains newly introduced parameters, which are C and D. The parameters A and B are almost identical in their trendability and monotonicity results as seen in 5.31 but the **A parameter** is slightly better. Visually, A and B do not show any obvious trend behaviour as seen in the plots in figure 5.30

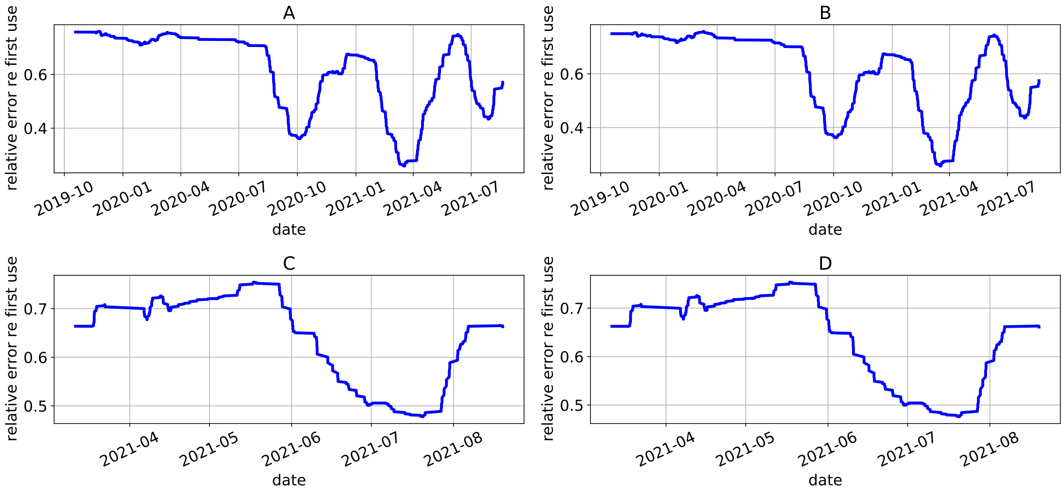


Figure 5.30: M2 Sub6 signals

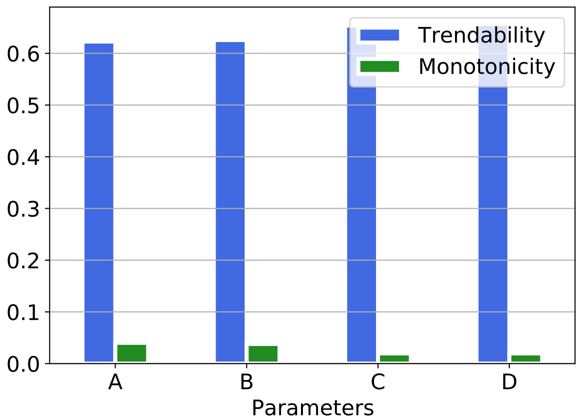


Figure 5.31: M2 Sub6 trendability and monotonicity analysis

This concludes the selection of the representative parameters for each subtest. This selection process attempts to optimize the amount of useful trend information and tries to reduce noise by removing parameters which show little to no trend behaviour.

Observations have shown that there is **no difference between ETT1 and ETT2 when selecting parameters with best trend behaviour**. Both show similar results; thus, no distinction is made and all the above presented graphs can be applied for the ETT2 test tower as well. But it must be noted that due to the very small trendability and monotonicity characteristics/values some parameters have been chosen rather arbitrarily as seen in the parameter selection process. Therefore, the similarity of the ETT1 and ETT2 test towers in regard to the trend behaviours of the parameters does not prove similarity in behaviour for the test towers. This means that the ETT1 and ETT2 could potentially be behaving differently, and the observed similarity is only a weak indication of their similarity in behaviour and should be handled with caution.

Result value types

Due to experience the results for phase measurements or THD measurements are the main result type for the subtests with multiple result types. Subtests with only one result type are not removed but analyzed using their single result type nonetheless. Therefore, the resulting subtests can be further reduced as seen in table 5.7 resulting in a total number of 15 subtests.

Module	Subtest	Number of Parameters	Channels	Types of Result Units	Sum of Subtest Runs
M1	Sub1	1	1	1	1
	Sub2	1	1	1	1
	Sub3	1	1	1	1
M2	Sub1	1	1	1	1
	Sub2	1	1	1	1
	Sub3	1	1	1	1
	Sub4	1	1	1	1
	Sub5	1	1	1	1
	Sub6	1	1	1	1
	Sub7	1	1	1	1
M3	Sub1	1	1	1	1
	Sub2	1	1	1	1
M4	Sub1	1	1	1	1
	Sub2	1	1	1	1
	Sub3	1	1	1	1
				Sum of subtests:	15

Table 5.7: Overview of subtests after removing parameter dependency and result types

Miscellaneous reasons

There are subtests that have not been discussed yet which are the subtests of the modules M3 and M4.

M3

The M3 subtests, both the Sub1 as well as Sub2 show the following issue as seen in 5.32 and in 5.33 respectively. This can be observed for multiple parameters, especially for larger parameters.
The figure 5.32 shows three distinct data point clouds.

Even though these subtests are not tested for multiple channels, there seem to be multiple distinct trends visible. This effect is therefore not due to channels and additional tests have shown that this effect can also be observed for other parameter

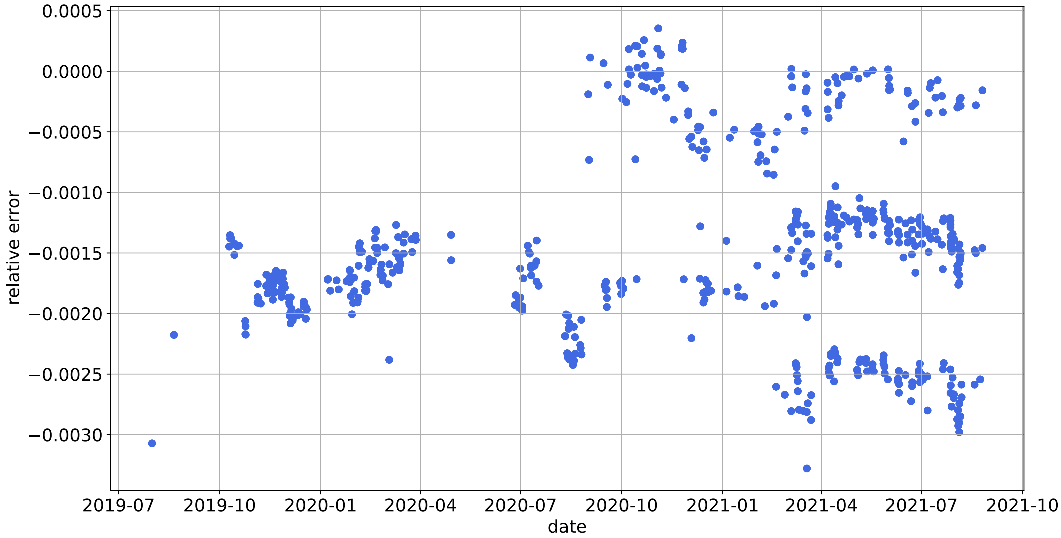


Figure 5.32: M3 Sub1

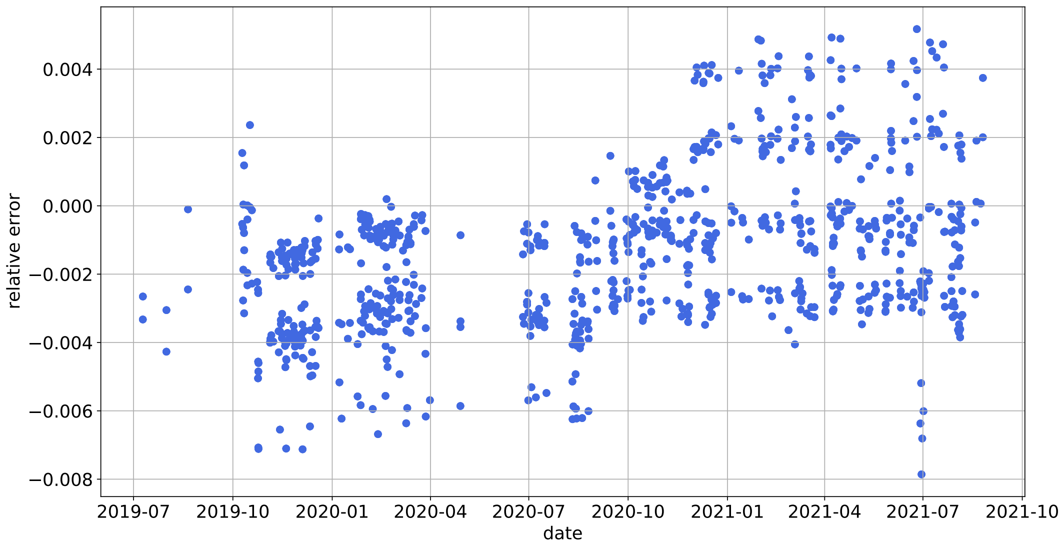


Figure 5.33: M3 Sub2

as well. A similar effect can be seen for the subtest Sub2 as shown in 5.33.
The exact reason for this type of behaviour could not be found which could be of interest and might be a potential future work. Thus, the subtests of M3 are discarded.

M4

A similar issue can be found for the M4 module subtests, to be exact for the M4 Sub1 and the M4 Sub2. The behaviour can be observed in figures 5.34 and 5.35.

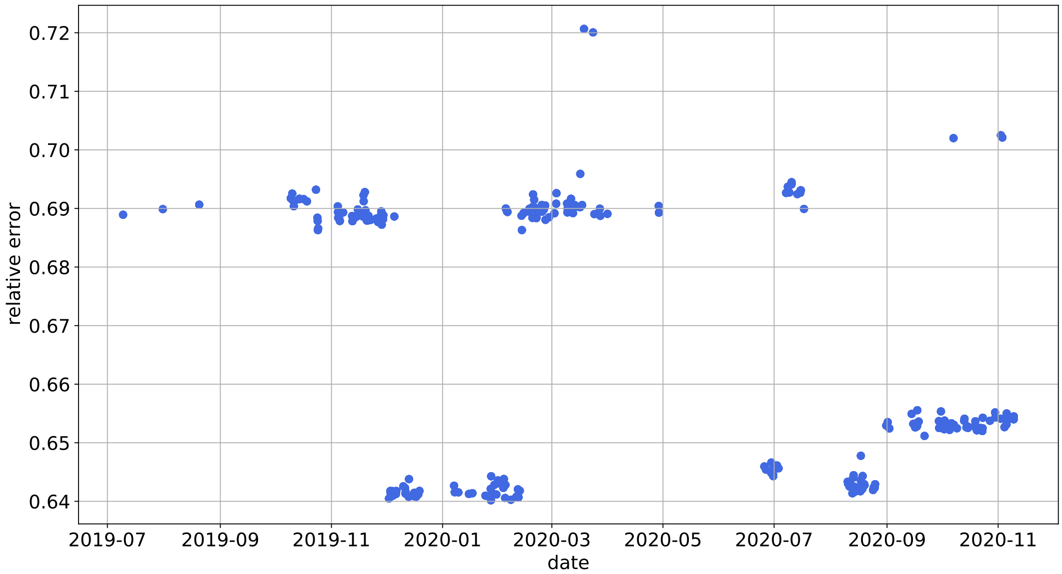


Figure 5.34: M4 Sub1

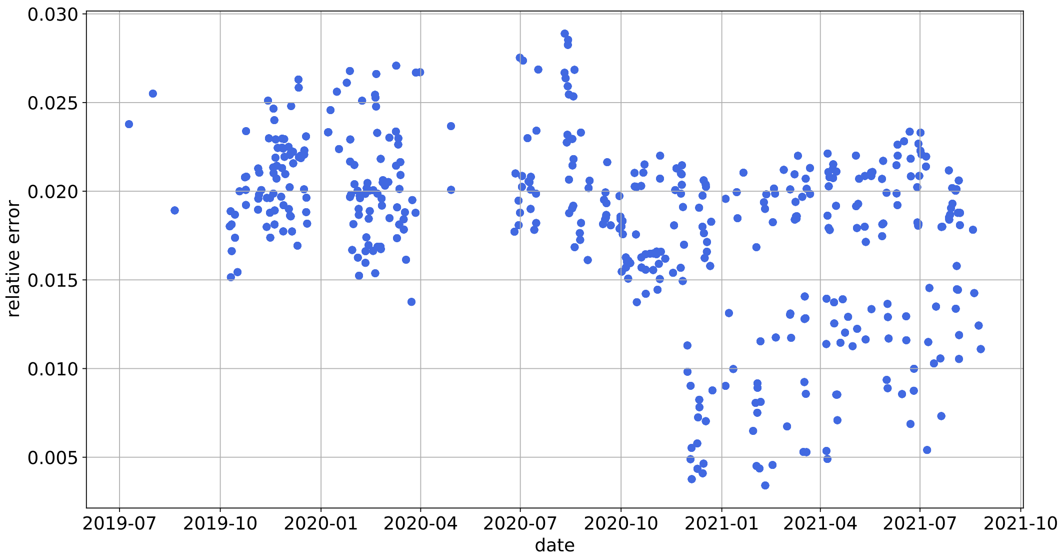


Figure 5.35: M4 Sub2

There seem to be some kind of jumps in the data, especially for the M4 Sub1 subtest

seen in figure 5.34. They can be potentially explained by the fact that there might have been hardware of software changes on the test tower. Unfortunately, these kinds of events are not well document and therefore an exact reason has not been determined yet. Similar jumps in the data can been also seen for the M4 Sub2 subtest illustrated in figure 5.35.
Since this kind of jumps disrupt the trend behaviour, the subtests of M4 Sub1 and Sub2 or not included in the final dataset. As a future work an investigative study could be conducted on these behaviours.
The subtest M4 Sub3 is also removed because its data could not be easily accessed using SQL queries. Since the fetched data is incomplete, this subtest is also removed.

5.8.2 Dimensionality reduction

Each subtest has multiple features which could potentially be used for the PdM when training a RUL model. Having too many features has two main problems.
Firstly, in case a ML model has to be trained, too many features lead unavoidably to long training times depending on the type of the model. Secondly, a ML model is sensitive to "bad" features which have little to no predictive ability. For a ML model that kind of features are only additional noise making the ML less accurate. Therefore, a dimension reduction is important.
Another important point is to consider the interpretability of the features. Since the selected PdM model for this research is likely to be a degradation model due to missing run-to-failure data, it is important to be able to create a threshold for a feature. If the behaviour of the feature cannot be interpreted, a threshold will be difficult to set. In case a ML model is actually implemented a dimensionality reduction will be performed using for example PCA or other techniques. The implementability of PdM is discussed in the following chapter.

5.9 Final dataset

The dataset started off with 259 subtests which contains redundant and partially useless data which can be considered noise. In the first step, the channel dependencies were summed up into a single channel since their trend behaviours were similar reducing the number of subtests to 100. The parameter dependency was handled by removing subtests with parameters which had low trendability and monotonicity metric values which indicated a weak trend behaviour. This resulted in subtests for each functionality to have only one parameter resulting in a total number of subtests to 15. Due to some miscellaneous reason, the subtests for the modules M3 and M4 were also removed resulting in a final total number of subtests of ten.
The final dataset can be seen in table 5.8.

A justified concern is that the reduction of the number of subtests reduces the size of the dataset which is contra productive since a bigger dataset is generally better for a data-driven modelling. But it must be noted that the removal of the channel

Module	Subtest	Number of Parameters	Channels	Types of Result Units	Sum of Subtest Runs
M1	Sub1	1	1	1	1
	Sub2	1	1	1	1
	Sub3	1	1	1	1
M2	Sub1	1	1	1	1
	Sub2	1	1	1	1
	Sub3	1	1	1	1
	Sub4	1	1	1	1
	Sub5	1	1	1	1
	Sub6	1	1	1	1
	Sub7	1	1	1	1
				Sum of subtests:	10

Table 5.8: Overview of the final subtests to be used for PdM

dependency did not lead to a reduction of the data size since the channels were simply summed up into one single channel. It did lead to a reduction of the number of subtests which is important because it shows that some of those subtests where redundant and could be summed up.
The parameter dependency on the other hand did reduce the data size. But here the argument is that by selecting the best parameter (i.e. parameter with the best trend metric results) and removing the worst parameters, the dataset left only the subtest with the most trend information and removed subtests with the least trend information(i.e. subtests which can be considered noise or unnecessary information). The removal of the M3 and M4 subtests is also justified since they contained data which had inexplicable trend behaviour. Such behaviour can also be classified as noise and should therefore be removed.

6 Data exploration

The first section performs general data exploration techniques on three subtests of the final dataset which gives insight into the dataset and its characteristics.
In the following sections the subtests are analyzed on the suitability for a PdM system. If a trend behaviour in a subtest is detected, it can be used to implement a degradation model as explained in chapter 1.5.
The final sections analyze the differences between the ETT1 and ETT2 tower as well as the differences between the model-driven reliability analysis and the data-driven trend analysis.

6.1 Exploration

The exploration depicts the major characteristics of the final dataset on the basis of three subtests. There is one subtest for the M1 module which is M1Sub2 and two subtests for the M2 module which are M2Sub3 and M2Sub7. To show some differences in data quality, the M2Sub3 is chosen as the subtest with the best trendability metrics values and M2Sub7 as its counterpart with the worst trendability metrics values. The exploration focuses on the analysis of the ETT1 test tower.
The behaviour of the other subtests is similar to the selected ones and are therefore not explicitly plotted and explained.

Figure 6.1 shows the initial data of the subtests after fetching from the DB or to be more exact the relative errors. It can be seen that each subtest differs in their value range and standard deviations. These subtests are fundamentally different because each of them tests a different functionality within the DUT thus there is no correlation between the subtests.

The different standard deviations of each subtest can be seen in figure 6.2. Each subtest shows different standard deviation value as well as different shapes in the distribution.

In can be seen that without a proper filter of the data there is no trend behaviour as seen in figure 6.1. This kind of data can also be classified as stationary. A time series data as given for the subtests is stationary if it does not any trend or seasonal characteristics. In order to have a fitting feature, the subtests have to be filter until the data shows signs of trend behaviour, meaning it is non-stationary. This can be tested using the **Augmented Dickey-Fuller test**[18]. For the test the null hypothesis is that a unit root exists in an autoregressive time series model. The test yields a **p value** where the following applies:

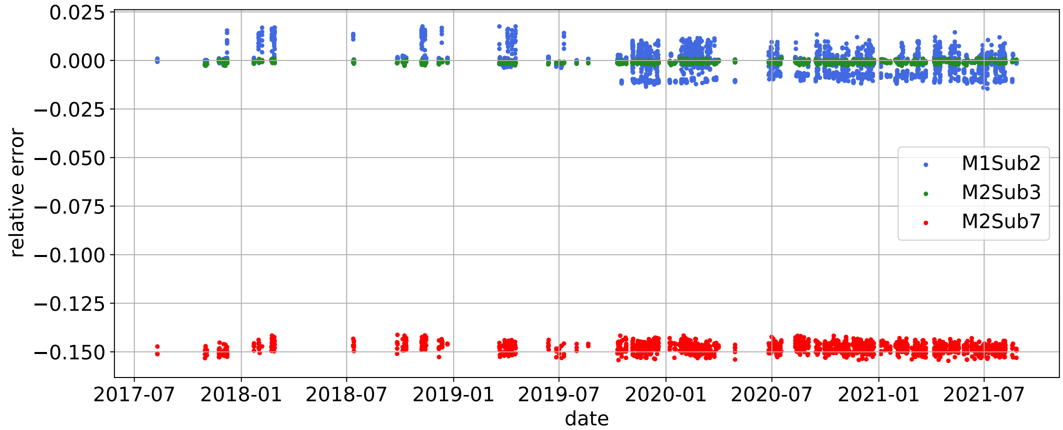


Figure 6.1: Relative error for the selected subtests

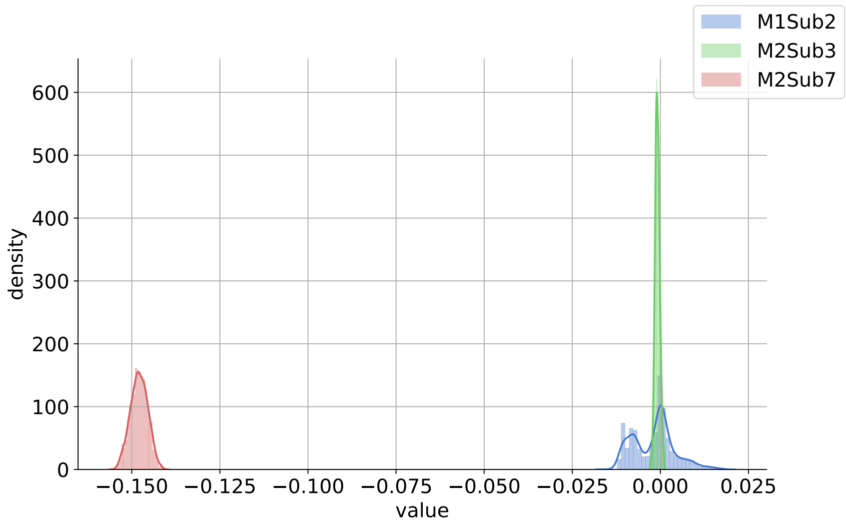


Figure 6.2: Standard deviation of relative error for the selected subtests

- **p-value $>$ 0.05:** Fail to reject the null hypothesis, the data has a unit root and is **non-stationary**.
- **p-value $<=$ 0.05:** Reject the null hypothesis, the data does not have a unit root and is **stationary**.

The filtering or smoothing of the data as explained in the chapter 5.6.6 returns the values shown in table 6.1. It shows the comparison of the filtered and unfiltered "relative error"-features.

It can be observed that the p-values for the original data points are all smaller that 0.05, meaning that the data is stationary therefore does not have any trend behaviour.

	p-value for original data	p-value for smoothed data
M1Sub2	0	0,706813
M2Sub3	0	0,142635
M2Sub7	0	0,01596

Table 6.1: Dickey-Fuller test result

After the smoothing process the p-value increases. M2Sub7 on the other hand shows such little trend behaviour that even extensive smoothing is useless.

The frequency domain analysis of the relative error does not show any noticeable characteristics as seen in figure 6.3. The analysis was performed using the Fourier transform. No specific frequency or harmonic stands out in the plot.
The analysis of time-frequency domain features as shown in chapter 5.6.9 have also been analyzed but the analysis did not yield any relevant information.

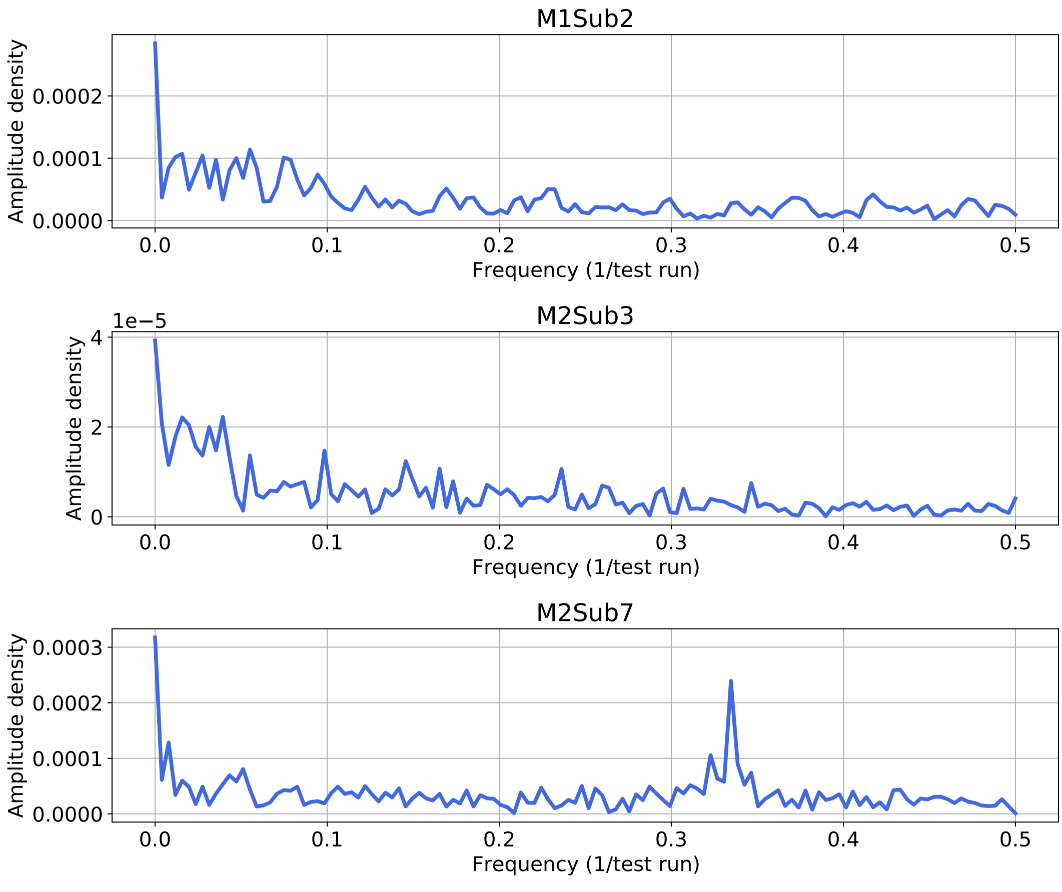


Figure 6.3: Frequency domain analysis with Fourier transform for the selected subtests

As explained in chapter 5.6.4 the “relative error re first use”-feature serves the purpose to make the subtests comparable. The illustration in figure 6.4 shows what effect the computation of the “relative error re first use”-feature has. The straight parts of the subtest at the beginning of the plots are the parts that were used for the window size of the smoothing filter. The same goes for all the other feature plots.

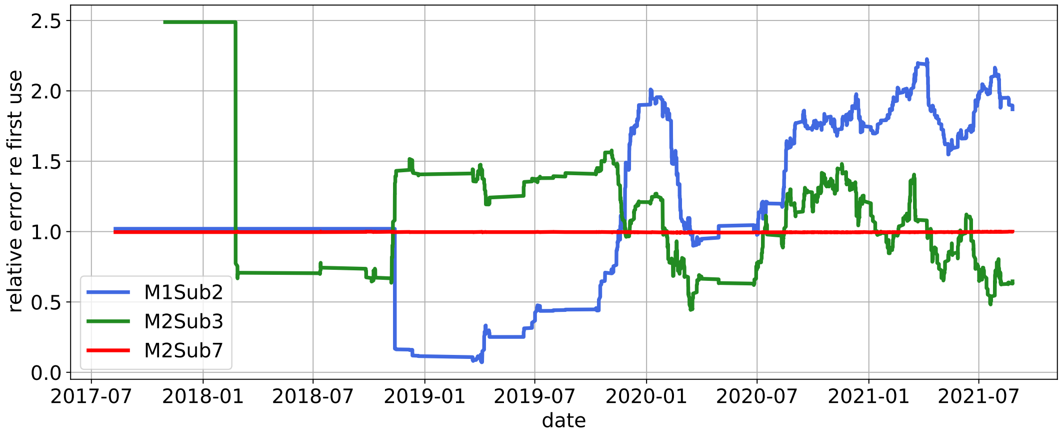


Figure 6.4: Relative error re first use for the selected subtests

It can be seen that the subtests M1Sub2 and M2Sub3 are now in a similar value range. The M2Sub7 shows value little trend behaviour and therefore appears as a straight line in the plot.

Additional time series features were introduced in chapter 5.6.7 in table 5.4. Since none of the additional features showed a promising trend behaviour throughout the entire dataset only a selected few features are discussed which are `Std`, `Skew` and `Kurtosis`. Figure 6.5 shows the standard deviation for each subtest. While M1Sub2 shows some trend behaviour, the other two subtests do not. The same goes for the remaining seven subtests where none shows a clear sign of a trend.

The illustration in figure 6.6 shows the skewness of the data. This feature does not show any trends either. Apart from the relatively large variance of the M2Sub7 subtest, the subtests do not show any particular characteristics. The kurtosis feature is depicted in figure 6.7. Apart from a slight falling tendency of the data there are no clear signs of a trend.

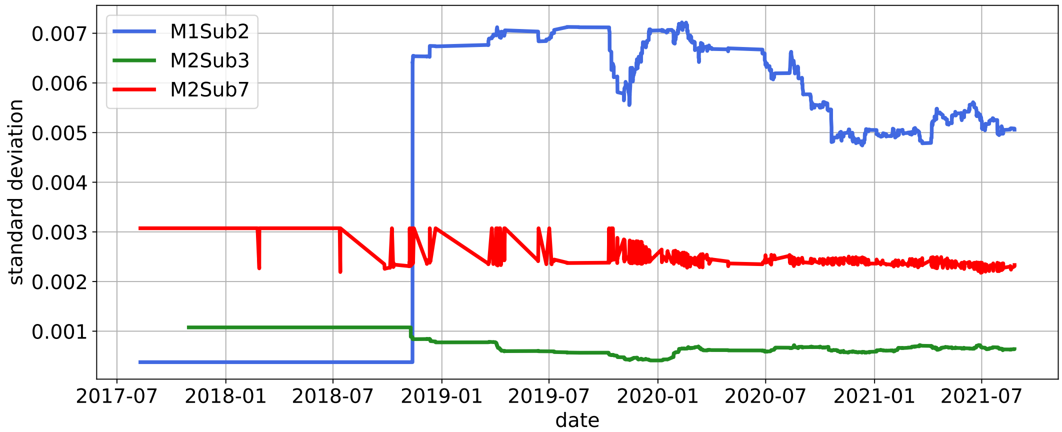


Figure 6.5: Standard deviation for the selected subtests

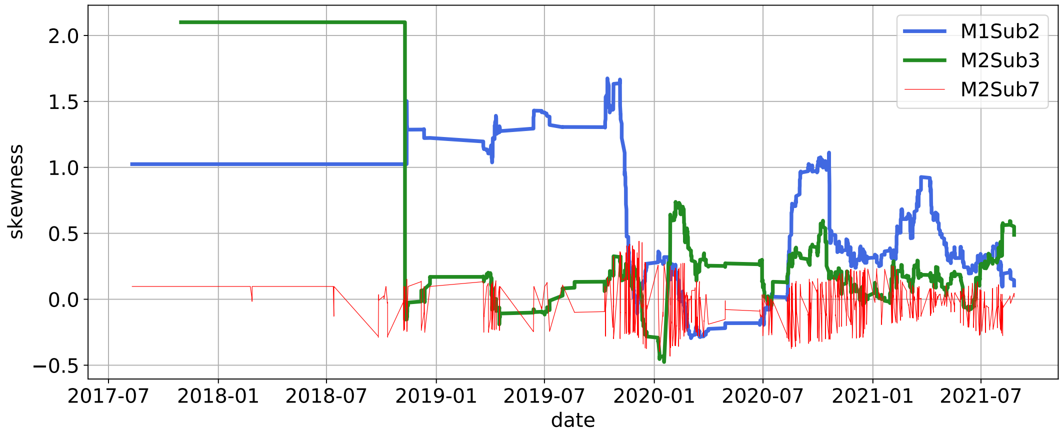


Figure 6.6: Skewness for the selected subtests

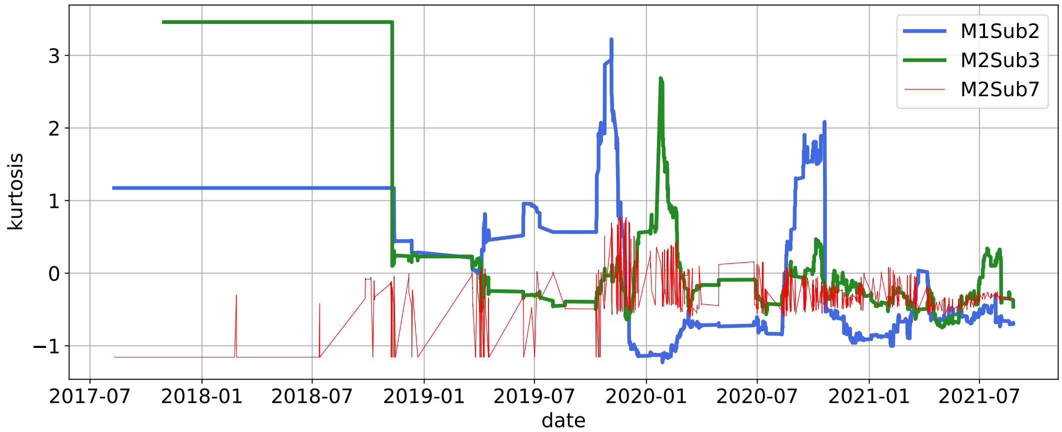


Figure 6.7: Kurtosis for the selected subtests

One possible effect for deterioration might be the **time that the test towers spend unused**. Thus, plotting the time points at which the test tower was unused for longer than a certain amount of time could reveal some new insight. Specific parts of the short-term trends could be due to this effect. In figure 6.8 it can be seen that the testing of the DUT usually happens in chunks. This means that for a specific time period there are many test runs as seen in the green framed data points which is then followed by a time period of no test runs.

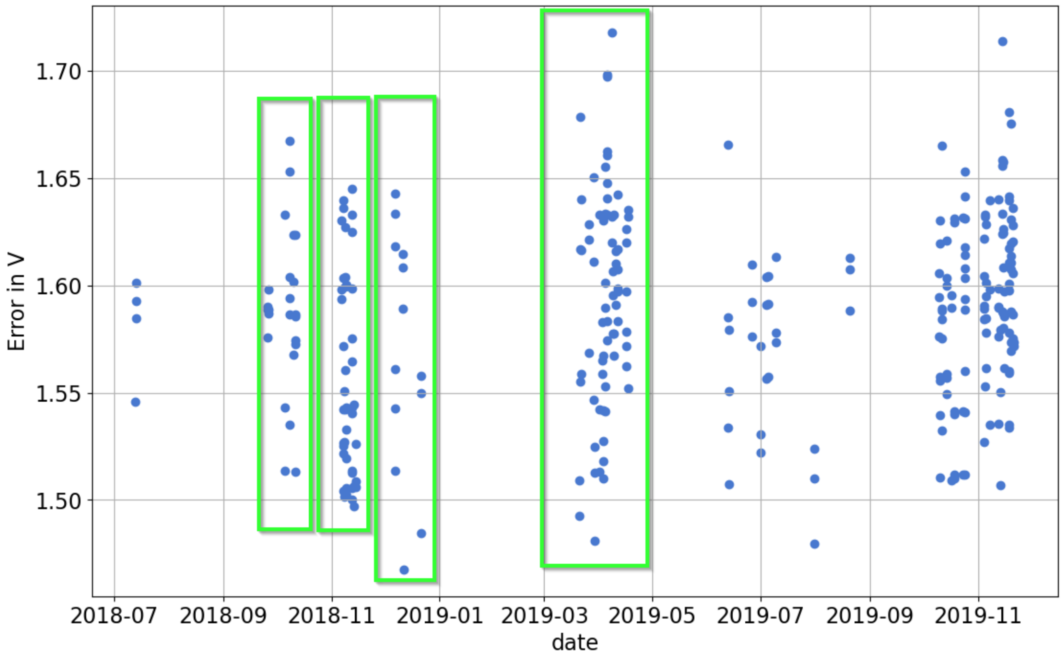


Figure 6.8: Testing of DUT happens in chunks

Figure 6.9 shows the relative error re first use including the time points at which the subtest was unused for 7 days. All the other points without the markers have all been ran within 7 days after the last test run. It can be observed that there is no clear correlation between a trend monotonicity change and the time points. The illustration in figure 6.10 shows the markers for an unused time duration of three weeks or 21 days. Even longer periods of unused subtests do not seem to affect the deterioration or the monotonicity change of the data. Thus, it can be concluded that the duration of “unusedness” does not affect the deterioration test tower strongly but a possible influence cannot be excluded.

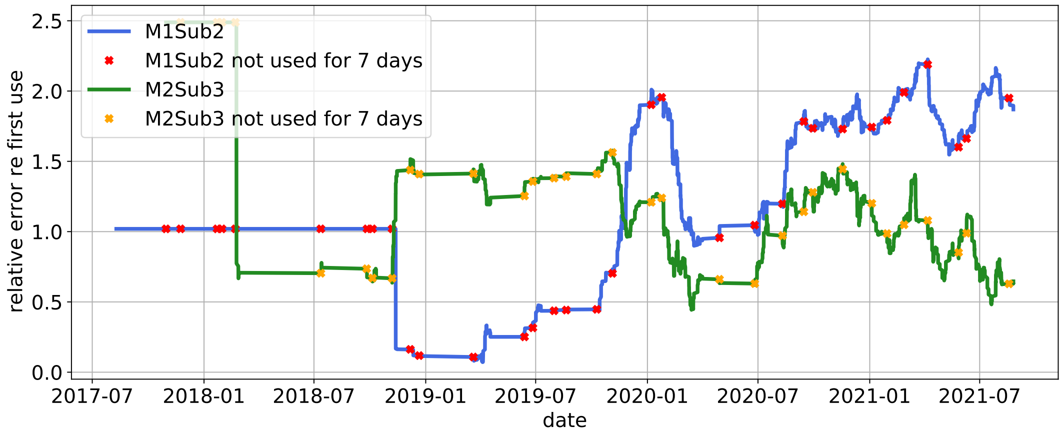


Figure 6.9: Relative error re first use including the time points at which the subtest was unused for 7 days

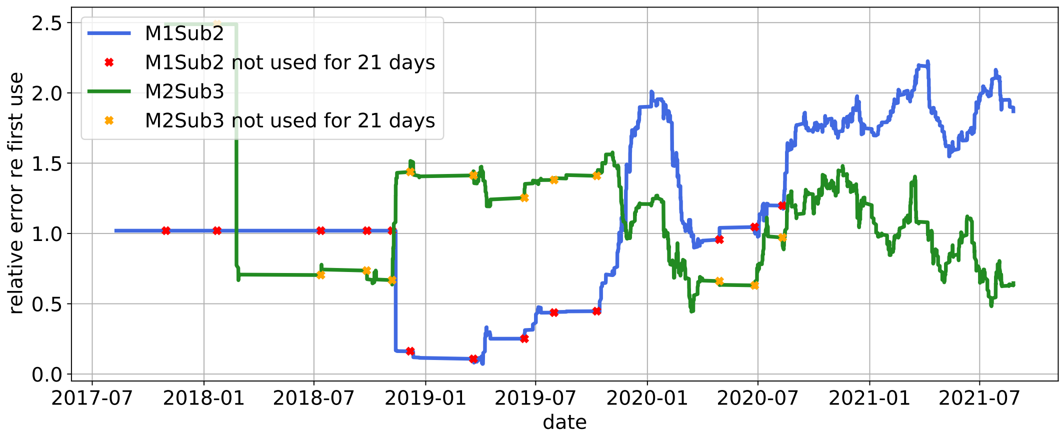


Figure 6.10: Relative error re first use including the time points at which the subtest was unused for 21 days

Other data exploration techniques were also used, such as the Autocorrelation, Partial Autocorrelation, correlation matrix for the time series data etc. But none of these technique yielded a relevant insight.

All time series features showed little to no trend behaviour and are therefore not suitable as condition indicator. The "relative error re first use"-feature does not meet requirement either but it has the best interpretability. Thus, the main condition indicator is the relative error re first use.

6.2 Trend analysis

In order to compare the trendability of the subtests, the trendability analysis tools are used on the "relative error re first use"-feature of each subtest.
Here are the results for ETT1 for the monotonicity analysis and the trendability analysis seen in figure 6.11.

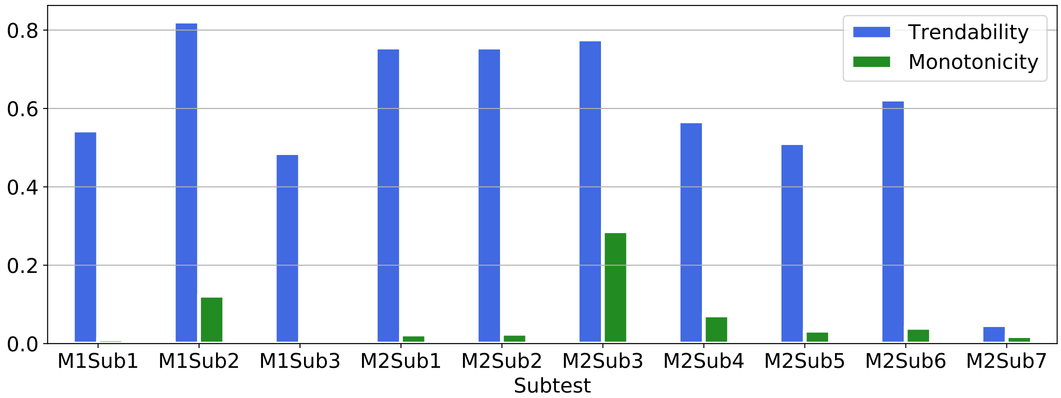


Figure 6.11: Comparison of subtests in monotonicity and trendability for ETT1

It can be seen that there are some subtests that show relatively strong trend behaviour. For monotonicity a $M > 0.3$ is desired otherwise the trend of the feature is not prevalent and it cannot be used for prognostic purposes.
The trendability is already strong for many subtests but anything below 0.5 should be discarded as noted in chapter 5.7.2.
The monotonicity values are all below the 0.3 threshold and thus do not meet the requirements. The only subtest which comes near the threshold is the M2Sub3 subtest.

Conclusively, it can be seen that none of the remaining subtests meet the requirements as a potential prognostic entity. Even if the trendability values are within requirements for some subtests, the **monotonicity values do not meet the threshold** to be considered a useful predictor. Thus, **it is advised against implementing a PdM for the test towers**.
Since no trend behaviour could be shown for any of the subtests, a RUL model for PdM cannot be implemented since it relies on the trend behaviour of the data. An anomaly detection as a PdM on the other hand could be possible since it does not rely on trend characteristics. In order to test this hypothesis, an anomaly detection model is implemented and tested.

6.3 ETT1 vs. ETT2

The ETT1 and ETT2 test towers are similar in their HW and SW composition but differ in their applications and their modes of operation. Thus, a proper comparison of the test towers and their data is not possible.

Apart from the fact that the towers are the same type of tower and are constructed equally, the ETT1 and ETT2 have huge differences in the way they are used and the timing when they were operated which make it difficult to compare.
The ETT2 has been the main tower for testing the CMC430 since its commissioning in 2017 until around 2019. Starting from 2019 the main testing tower for CMC430 became the ETT1. The following bullet points list the main differences:

- ETT2 was used the first 2 years since commissioning when the tower was still in its commissioning and development/adjustment phase. There were more software and hardware adjustments.
- Since there have been more changes in HW and SW, there are more error sources for the measurements which cannot be traced easily since the documentation is lacking.
- ETT1 is mainly used since 2019 after bigger faults and flaws have been fixed.
- The usages of the other DUT - ARCO400, have not been analyzed, thus those potential effects cannot be taken into account either. Meaning, the ARCO400 subtests also use the ETT1 or ETT2 test tower for their test which can have an impact on the test towers and their potential degradation.

In figure 6.12 are the results for the trend analysis for the ETT2 test tower.

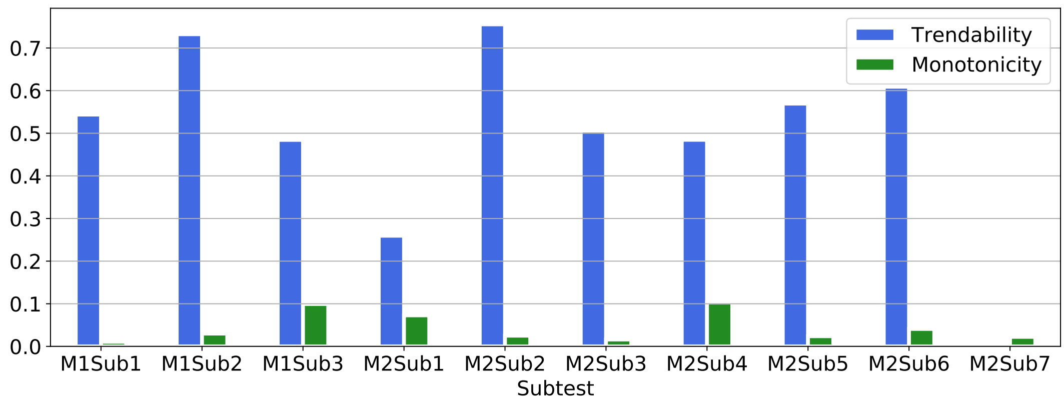


Figure 6.12: Comparison of subtests in monotonicity and trendability for ETT2

It can be seen that it shows the same characteristics as for the ETT1 results. While the trendability values are relatively high the monotonicity results are way below the

necessary threshold.
The bar plot in figure 6.13 shows the comparison of the trendability results. It can be observed that most of the subtests have similar trendability values. The subtests M1Sub1, M2Sub1 and M2Sub3 show conflicting results where one test tower has a value within the threshold while the other is below it.

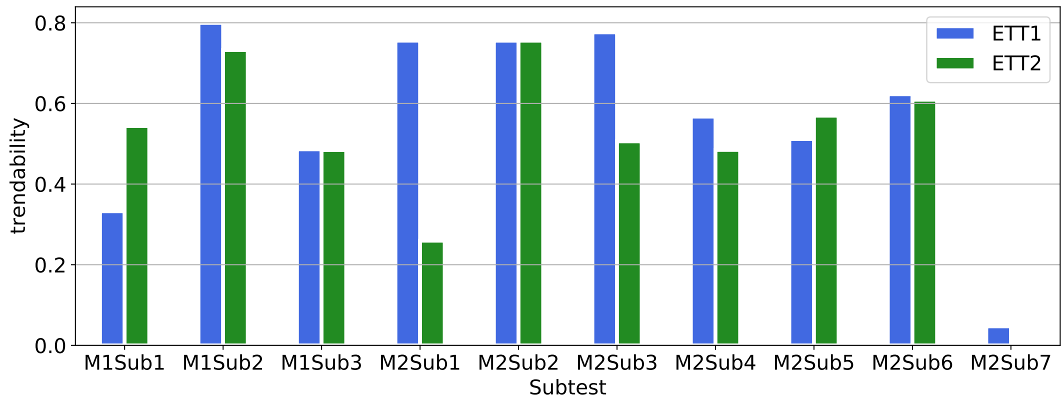


Figure 6.13: Comparison of ETT1 and ETT2 in their trendability

The comparison of the monotonicity analysis results are shown in figure 6.14.

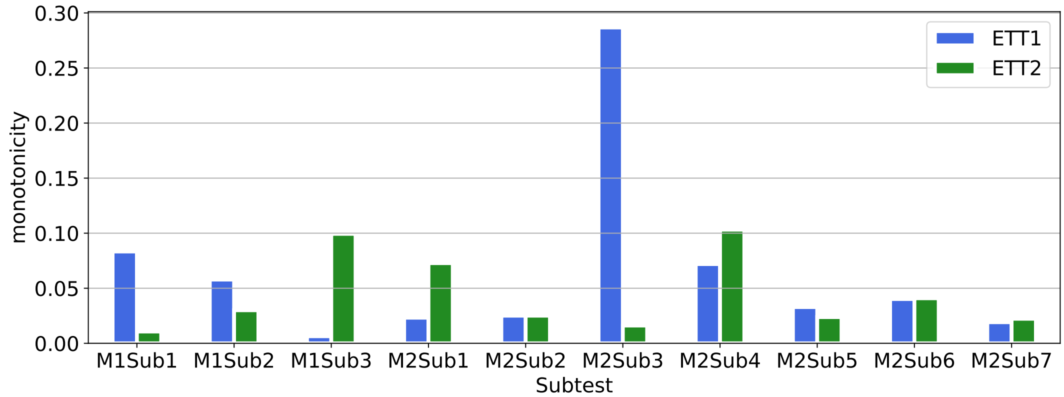


Figure 6.14: Comparison of ETT1 and ETT2 in their monotonicity

Since the significance of the monotonicity for such values in not given, it is hard to compare the results. After all, monotonicity values below the threshold are both weak trend behaviour indicators which means that those results can represent any non-trending time series. If only the numerical value is taken into account, the monotonicity results for the test towers are still quite different for a number of subtests, especially for M2Sub3. This is a subtest which showed the best trend behaviour for the ETT1 among all subtests.

If the errors and the apparent trends in the subtests are actually caused by the modules of the test tower, there might be a correlation between ETT1 and ETT2. However, a meaningful comparison is not possible since the operation time and the style of operation of the towers are different.

Since ETT1 is the currently used test tower for the CMC430, it is used as the default test tower for all the following analyses. Meaning, the comparison of the modelling approaches and the implementation of the anomaly detection all use the dataset of the ETT1 test tower.

6.4 Trend analysis vs. Tower Modelling

Since there are differences between ETT1 and ETT2, it is difficult to compare the modelling results to the data analysis results. The question arises whether there are any correlations between the modelling and the data analysis at all because the data analysis shows different results depending on the test tower.

The result of the reliability modelling roughly is proportional to the number of relays used in a subtest. This approximation can be done because all the failure rates λ are similar values. The number of relays for each subtest can be seen in table 6.2.

Subtest	Number of relays
M1 Sub2	5
M2 Sub1	17
M2 Sub2	17
M2 Sub3	10
M2 Sub4	10
M2 Sub5	17
M2 Sub6	17

Table 6.2: Number of relays for each subtest

In order to compare the reliability model and the subtest trends, the trends are approximated using a simple **linear regression** which yields a value for its slope. An example of a linear regression approximation of the trend can be seen in figure 6.15. It shows the relative error re first use for the M2Sub3 subtest and its linear regression.

The results of the regression for each subtest can be seen in table 6.3. Since it is not relevant whether the slope is rising or descending, the slope values in the table are absolute values.

Using the slope value, it can determine which one of the subtests has a larger slope thus an apparent larger degradation.
The comparison of the models is the comparison of their respective slopes or to be more exact the ranking of the slopes to the ranking of the reliability modelling results.

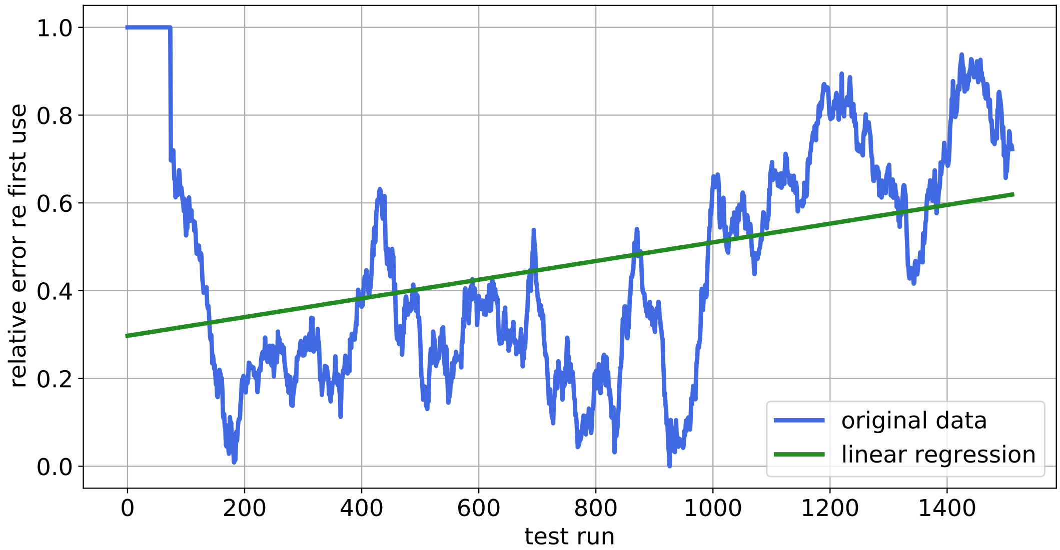


Figure 6.15: Linear regression of the M2Sub3 subtest

Subtest	Slope of linear regression
M1 Sub2	0,04693
M2 Sub1	0,06645
M2 Sub2	0,067079
M2 Sub3	0,394122
M2 Sub4	0,343886
M2 Sub5	0,182547
M2 Sub6	0,420201

Table 6.3: Value of slope for each subtest

The assumption is that there is an effect of the relays on the trend of the subtests. If that is the case, their effect should be proportionally stronger in subtests with more relays. The ranking for both models is shown in table 6.4.

The ranking shows the ranking of the degradation. The higher the ranking the stronger the apparent degradation.
It can be observed that the module M1 subtest with the least number of relays has the weakest degradation for both modelling approaches. Since the M1 module is a voltage amplifier where the relays are mainly switched without a load, this is very plausible. The subtests for the M2 module which is a current amplifier show higher degradations but there does not seem to be an actual correlation between the number of relays and the degradation observed in the data. But it should be noted that the linear regression approach is a very simple approximation of the trend and there is a possibility that this approach is not suitable.

Ranking of reliability	Subtest		Ranking of linear regression	Subtest
1	M2Sub1		1	M2Sub6
1	M2Sub2		2	M2Sub3
3	M2Sub5		3	M2Sub4
3	M2Sub6		4	M2Sub5
5	M2Sub3		5	M2Sub2
5	M2Sub4		6	M2Sub1
7	M1Sub2		7	M1 Sub2

Table 6.4: Ranking of reliability and linear regression

But there are some indications that the relays might have an effect(e.g. the M1 module shows the weakest degradation for both models). However, with only seven subtests which show some kind of trend (excluding all irregularities in the trend), this claim has only a weak basis. There are multiple reasons why the effects of the relays cannot be proven:

- **Too small number of subtests:** With only seven subtests the statistical relevance is too little. Additionally, not all subtests correlate with the reliability model. Thus, more subtests are needed to show a certain effect of the relays on the degradation.

- **Effects from other components:** Since the reliability analysis has the assumption that only the relays have an impact on the degradation, other component are ignored. But the results show that the effects of the other components potentially overweighs the effect of the relays.

- **Suitability of feature:** The main feature used was the “relative error re first use”-feature. It was proven to be useful for comparing results of multiple subtests but the trendability metrics have shown that it lacks in predictive or prognostic utility. This is not a flaw of the feature but a system condition. The current dataset of the test towers does not show any meaningful trends that can be linked to the effects of relays.

- **Not enough data:** In order to increase the number of subtests with trends, more data points are needed. Meaning, not the number of subtests is important but the number of test runs. The degradation process intensifies over time and therefore the trends will be more apparent after performing more tests. Thus, the current dataset with only four years worth of data might be not enough.

7 Implementation of PdM

Even though the monotonicity and trendability values for all subtests do not meet the requirements for a good predictive utility as shown in chapter 6.2, an implementation of a PdM system is still attempted, in particular the anomaly detection is implemented and evaluated.

7.1 Data preparation for ML

The given dataset with its preparations is already suitable for the ML models. Some additional preparations include data transformations such as using the RobustScaler[40]. The feature used for the RUL model is the "relative error re first use"-feature while the anomaly detection takes the original data, meaning the "absolute error"-feature or the "relative error"-feature. The subtest which is used for training is the M2Sub3 subtest since it showed the best trend behaviour.

7.2 RUL prediction

Since it was shown that none of the subtest in the final dataset meet the requirements for a suitable trend behaviour a working RUL ML model is difficult to implement. Two different ML models have been trained on the given dataset using the M2Sub3 subtest. The used ML models are the **LSTM network** and the **ARIMA model**.
One of the more modern techniques often used in the time series prediction is the **LSTM network**. The LSTM, short for *Long Short Term Memory*, is a popular and modern ML model used for different tasks such as forecasting of time series data. For information about LSTM see [43][21].
The training as well as the prediction show relatively big inaccuracies thus the RUL model with LSTM was dropped.
A more classical ML solution is the ARIMA model. ARIMA, short for *Auto Regressive Integrated Moving Average* is a class of models that describes a time series using its past values such as its lags and the lagged forecast errors. This allows to forecast future values [9][21].
Similarly to the LSTM model, the ARIMA model showed weaknesses in accurately predicting the RUL.

The failed RUL models verify the proof that the dataset has insufficient quality due to missing trend characteristics in the data.

7.3 Anomaly detection

Another task of PdM is to be able to notice unusual operations of a system. In order to predict those, it is imperative to be able to detect anomalies. An anomaly detection might be able to perform well even without trendable data by using other characteristics of the dataset. Those other characteristics could be detected by using ML models.
The anomaly detection for this research is implemented using the following ML techniques: **Interquartile Range (IQR)**[33] and **K-Means Clustering**[38][].

Some of the subtests contain outliers in their measurements. The **goal of the anomaly detection** is to be able to predict ahead of time before those outliers happen. The assumption is that there might be indications to an outlier-event in the previous data-points.
In order to test this idea, the “M2Sub3” subtest is used. It contains some outliers.
In figure 7.1 the result for IQR anomaly detection is depicted. The algorithm finds only the actual outliers. There does not seem to be an initial phase which might indicate to a coming outlier.

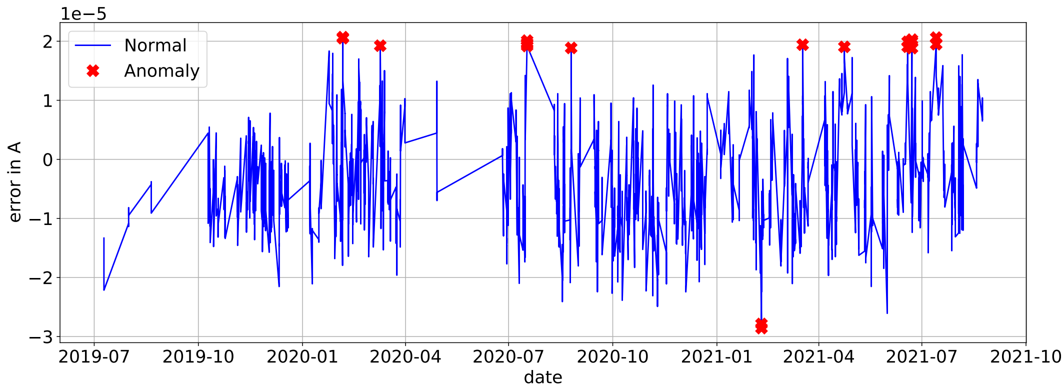


Figure 7.1: Anomaly detection with IQR

The implementation of the IQR anomaly detection is shown in listing 7.1. The IQR uses quantiles to evaluate whether a value is classified as an anomaly or not as seen in line 6.

```
1 current_feature = 'Error'
2 data_df[current_feature]
3 df = data_df.copy()
4
5 # Calculate IQR for the 1st principal component (pc1)
6 q1_pc1, q3_pc1 = df[current_feature].quantile([0.25, 0.75])
7 iqr_pc1 = q3_pc1 - q1_pc1
8
9 # Calculate upper and lower bounds for outlier for pc1
```

```
lower_pc1 = q1_pc1 - (1.5*iqr_pc1)
upper_pc1 = q3_pc1 + (1.5*iqr_pc1)

# Filter out the outliers from the pc1
df['anomaly_phase'] = ((df[current_feature]>upper_pc1) | (df[
  current_feature]<lower_pc1)).astype('int')
```

Listing 7.1: IQR implementation source code

A similar result can be observed for the K-Means algorithm shown in 7.2 and the source code for the implementation is seen in listing 7.2.

```
from sklearn.cluster import KMeans

df = data_df.copy()
kmeans = KMeans(n_clusters=2, random_state=42)
kmeans.fit(df[current_feature].values.reshape(-1, 1))
labels = kmeans.predict(df[current_feature].values.reshape(-1, 1))
unique_elements, counts_elements = np.unique(labels, return_counts=True)
clusters = np.asarray((unique_elements, counts_elements))
# Write a function that calculates distance between each point and the
  centroid of the closest cluster

def getDistanceByPoint(data, model):
    """ Function that calculates the distance between a point and
  centroid of a cluster,
            returns the distances in pandas series"""
    distance = []
    for i in range(0,len(data)):
        Xa = np.array(data.loc[i])
        Xb = model.cluster_centers_[model.labels_[i]-1]
        distance.append(np.linalg.norm(Xa-Xb))
    return pd.Series(distance, index=data.index)

# Assume that 1% of the entire data set are anomalies
outliers_fraction = 0.007

# get the distance between each point and its nearest centroid. The
  biggest distances are considered as anomaly
distance = getDistanceByPoint(df[current_feature], kmeans)

# number of observations that equate to the 13% of the entire data set
number_of_outliers = int(outliers_fraction*len(distance))

# Take the minimum of the largest 13% of the distances as the threshold
threshold = distance.nlargest(number_of_outliers).min()

# anomaly1 contain the anomaly result of the above method Cluster (0:
  normal, 1:anomaly)
df['anomaly_phase'] = (distance >= threshold).astype(int)
```

Listing 7.2: KMeans implementation source code

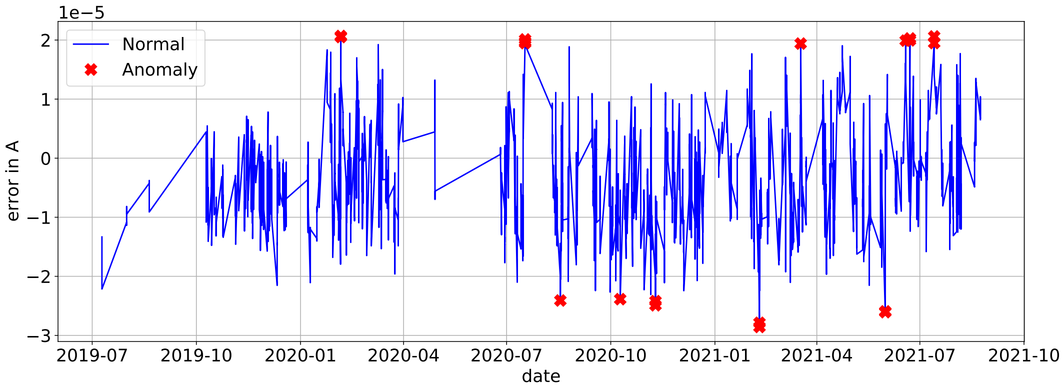


Figure 7.2: Anomaly detection with K-Means

There does not seem to be a correlation between the outlier and its previous data points. The ML models only find the outliers themselves, but no anomalies are found immediately before the anomaly occurs. If the ML model could classify data points before the outlier as an anomaly, it would be possible to use the characteristic to predict an outlier. Since this is not the case, as it can be seen on the plots, the anomaly detection does not work.

Thus, it can be concluded that neither a RUL prediction nor an anomaly detection can be implemented for the test tower.

8 Proposed further action for PdM

The tower itself has been proven to be very accurate or not noticeably prone to degradation for the last approximately four years. The given data was strongly limited and was partially insufficient thus with a larger dataset, different trends and degradations could be found. Therefore, it is recommended to analyze the test tower data after additional 5-10 years which will result in a dataset more than double the current size. Compared to the currently available data size, it should increase the data quality and should yield additional insights.
If the measurement error of the subtests has an exponential growth, the trend behaviour will be even more pronounced in 5-10 years time.
Some **proposed actions for the test tower** are the following:

- If possible, a standardized test runs of a DUT with designated conditions. This should happen at least once every week in order to have more data points.
- Documentation of each maintenance activity and occurring issues on the tower including proper classes of error types.
- Defined ETL process for easier access of data. The current JSON formats are not consistent and differ in format.

The dataset of the test tower is large and therefore, many more analytical analyzes can be performed. This work focuses only on time domain features and features which are easy to interpret leaving a lot of room for additional analyzes.
A rough analysis of the subtests using **Fourier transform** did not show any particular frequency information but this aspect could be inspected in more detail. **STFT** and **Wavelet transform** on the subtests did not show any obvious characteristics. But since this type of analysis is full of potential it could be used to analyze different time series features such as the features listed in table 5.4. Each of those features can yield new information which could potentially be used for a PdM.

Apart from the test towers, a PdM system can be implemented **for other OMICRON devices** as well. Ideally, the devices are part of the commonly used applications shown in figure 1.2. If that is not the case, the system identification and the search for condition indicators have to happen manually. This often requires a lot of work and time.
Moreover, the data quality and the data size have to be suitable. The data quality means the existence of a condition indicator which captures the trend behaviour of the system well enough. This can be evaluated using the trendability and monotonicity

metrics. The data size on the other hand depends on the type of the system and its deterioration and trend behaviour. If the deterioration is strong or if it follows a clear characteristic, the data size can potentially be smaller. For very small changes in the trend a lot of data is required to capture the behaviour accurately.

The proposal for OMICRON devices is to capture as many data points as possible since the amount of data points is always beneficial for a possible future data-driven model.

It is beneficial if the data gathering process of the OMICRON devices includes more than one feature. The best solution would be to include multiple features from different domains, e.g. the error of a module, the temperature, the humidity etc. Those features could show correlations which could later be used as condition indicators.

9 Conclusion

This research aimed to identify whether an implementation of a PdM strategy for the test towers at the company OMICRON electronics GmbH is possible. Based on the reduced and optimized dataset it was shown that the trendability and monotonicity metrics do not meet the required threshold. Thus, it could be proven that a PdM implementation for the given test towers is unnecessary and not possible.
Since the trend behaviour was not given, the RUL implementation was not implemented. An additional analysis on the implementability of a potential anomaly detection was conducted with the assumption that a ML model could find an anomaly without relying on the trend behaviour. This hypothesis was observed to be inaccurate since the anomaly detection with two different ML models could not predict anomalies ahead of time.
Thus, the ideal solution which was mentioned in the chapter 2 could not be achieved. It was not possible to predict RUL with the given data quality thus a prediction of the location of a possible failure cannot be made either.

The size of the given dataset was large and needed a restructuring by reducing and summarizing the subtests. The number of test runs for the last four years amount to just around 500 but the number of subtests with their different channels, result types and parameters make the data size increase strongly. A large amount of effort was poured into structuring the dataset and optimizing it down to ten essential subtests by removing redundant and noisy data. Therefore, many potential analysis types were not used on the dataset and some subtests remained unexplored which can be resumed in a potential future work.

Another possible **future work** could be done in the next 5-10 years when more data is available. The more data is accumulated over time, the more likely it is to detect a possible degradation of a system consequently showing signs of degradation which could not be found in the research.
It could also be interesting to **analyze the "calibration"-subtests** as shown in figure 3.6. Theoretically, those subtests should not contain any degradation trends since the influence of the tower is removed in the "adjustment"-subtests but that is only the theory. A proper scientific analysis would give more insight.
The tower also contains the DMM HP3458 as part of its equipment which is also calibrated on a regular basis. Those calibration datasets are also stored in the database. There are too little data points at the moment (approximately 50) but in ten year's time, there should be around 200 values which could paint a better picture of the entire test tower black box system. There could be correlations between measurement

trends and calibration trends which could be taken into account for a possible future PdM.
This work had its main focus on the adjustment test subtest. But it is possible to use the calibration subtests as well by **using the Golden Device**. This is a device which is never adjusted, meaning it stays the same. By using this to one's advantage it should be possible to compare the two test towers more accurately. The only downside is that the Golden Device is not run often. For the last four years there have been 50 test runs on the ETT1 which is a very small number and is rather insufficient for a potential ML model.
It should also be noted that the modules M3 and M4 were not included in the final dataset of this research due to their inexplicable trends in the data. This can be investigated in more detail.
Since many maintenance services are currently not being documented, it could be a good idea to create a system which allows an easy registration of a maintenance activity. Such a system could offer dropdown menus for selecting a type of fault and a comment section. Using such information could potentially allow a classification ML model to be implemented and used for real-time classification of new errors and failures.

Bibliography

[1] *104-1-A-24-1D Relays Datasheet pdf - Reed Relays. Equivalent, Catalog.* URL: https://datasheetspdf.com/pdf/1266606/pickering/104-1-A-24-1D/1 (visited on 08/21/2021).

[2] *104-2-A-24/2D.* en-GB. URL: https://www.pickeringrelay.com/product/104-2-a-24-2d/ (visited on 08/21/2021).

[3] *2.5. Decomposing signals in components (matrix factorization problems) — scikit-learn 0.24.2 documentation.* URL: https://scikit-learn.org/stable/modules/decomposition.html#pca (visited on 08/22/2021).

[4] Koceila Abid, Moamar Sayed Mouchaweh, and Laurence Cornez. "Fault Prognostics for the Predictive Maintenance of Wind Turbines: State of the Art". en. In: *ECML PKDD 2018 Workshops.* Ed. by Anna Monreale et al. Vol. 967. Series Title: Communications in Computer and Information Science. Cham: Springer International Publishing, 2019, pp. 113–125. ISBN: 978-3-030-14879-9 978-3-030-14880-5. DOI: 10.1007/978-3-030-14880-5_10. URL: http://link.springer.com/10.1007/978-3-030-14880-5_10 (visited on 07/22/2021).

[5] *alfg2 datasheet, alfg2 pdf, alfg2 Data Sheet PDF - Datasheetspdf.com.* URL: https://datasheetspdf.com/datasheet/search.php?sWord=alfg2 (visited on 08/21/2021).

[6] alldatasheet.net. *PB114024 Datasheet(PDF) - Tyco Electronics.* en. URL: https://www.alldatasheet.net/datasheet-pdf/pdf/409047/MACOM/PB114024.html (visited on 08/21/2021).

[7] *ARCO-400-Brochure-ENU-1.pdf.* URL: https://www.omicronenergy.com/download/document/3929A0A7-118E-4E25-98C6-11C6150463AA/ (visited on 08/28/2021).

[8] M Bevilacqua and M Braglia. "The analytic hierarchy process applied to maintenance strategy selection". en. In: *Reliability Engineering & System Safety* 70.1 (Oct. 2000), pp. 71–83. ISSN: 0951-8320. DOI: 10.1016/S0951-8320(00)00047-8. URL: https://www.sciencedirect.com/science/article/pii/S0951832000000478 (visited on 07/27/2021).

[9] *Chapter 8 ARIMA models | Forecasting: Principles and Practice (2nd ed).* URL: https://Otexts.com/fpp2/ (visited on 07/30/2021).

[10] *CMC-430-Brochure-ENU.pdf.* URL: https://www.omicronenergy.com/download/document/498B6B23-61E9-4E38-BFA5-C002E5826036/ (visited on 08/28/2021).

[11] *Cost of Data Center Outages.* Feb. 2016. URL: https://datacenterfrontier.com/white-paper/cost-data-center-outages/ (visited on 07/27/2021).

[12] *DDEController.pdf.* URL: https://www.te.com/commerce/DocumentDelivery/DDEController?Action=srchrtrv&DocNm=Definitions_Relays&DocType=SS&DocLang=EN (visited on 08/21/2021).

[13] *DSP2A-xxxxV Relay Datasheet pdf - Power Relay. Equivalent, Catalog.* URL: https://datasheetspdf.com/pdf/541328/Panasonic/DSP2A-xxxxV/1 (visited on 08/21/2021).

[14] Stefan Eberlin and Barbara Hock. *Zuverlässigkeit und Verfügbarkeit technischer Systeme.* de. Wiesbaden: Springer Fachmedien Wiesbaden, 2014. ISBN: 978-3-658-03572-3 978-3-658-03573-0. DOI: 10.1007/978-3-658-03573-0. URL: http://link.springer.com/10.1007/978-3-658-03573-0 (visited on 08/28/2021).

[15] "FEMP Operations & Maintenance Best Practices, Release 3.0". English. In: *Federal Energy Management Program* (Aug. 2010), p. 321. URL: https://www.wbdg.org/FFC/DOE/DOECRIT/femp_omguide.pdf (visited on 04/07/2021).

[16] *g2rl datasheet, g2rl pdf, g2rl Data Sheet PDF - Datasheetspdf.com.* URL: https://datasheetspdf.com/datasheet/search.php?sWord=g2rl (visited on 08/21/2021).

[17] Xiang Gong and Wei Qiao. "Current-Based Mechanical Fault Detection for Direct-Drive Wind Turbines via Synchronous Sampling and Impulse Detection". In: *IEEE Transactions on Industrial Electronics* 62.3 (Mar. 2015). Conference Name: IEEE Transactions on Industrial Electronics, pp. 1693–1702. ISSN: 1557-9948. DOI: 10.1109/TIE.2014.2363440.

[18] Peter G. Hackl. "Peter Winker: Empirische Wirtschaftsforschung und Ökonometrie: Springer, Berlin, 2.Auflage, 2007, xiv + 338 pp, US $34.95, GB £18.04, €24,95, ISBN 978-3540367789". de. In: *Statistical Papers* 52.4 (Nov. 2011), pp. 989–990. ISSN: 0932-5026, 1613-9798. DOI: 10.1007/s00362-010-0324-7. URL: http://link.springer.com/10.1007/s00362-010-0324-7 (visited on 08/28/2021).

[19] *IM02GR : Axicom Standard Signalrelais 2 Form C, 2 CO Kont.* de-de. URL: https://www.te.com/deu-de/product-1462037-9.html (visited on 08/28/2021).

[20] Kamran Javed et al. "Enabling Health Monitoring Approach Based on Vibration Data for Accurate Prognostics". en. In: *IEEE Transactions on Industrial Electronics* 62.1 (Jan. 2015), pp. 647–656. ISSN: 0278-0046, 1557-9948. DOI: 10.1109/TIE.2014.2327917. URL: http://ieeexplore.ieee.org/document/6824783/ (visited on 07/22/2021).

[21] Hui Liu. *Non-intrusive Load Monitoring: Theory, Technologies and Applications.* en. Singapore: Springer Singapore, 2020. ISBN: 9789811518591 9789811518607. DOI: 10.1007/978-981-15-1860-7. URL: http://link.springer.com/10.1007/978-981-15-1860-7 (visited on 08/28/2021).

[22] Alfred Mertins. *Signaltheorie: Grundlagen der Signalbeschreibung, Filterbänke, Wavelets, Zeit-Frequenz-Analyse, Parameter- und Signalschätzung.* de. Wiesbaden: Springer Fachmedien Wiesbaden, 2020. ISBN: 978-3-658-29647-6 978-3-658-29648-3. DOI: 10.1007/978-3-658-29648-3. URL: http://link.springer.com/10.1007/978-3-658-29648-3 (visited on 08/28/2021).

[23] Amanda Iglesias Moreno. *Moving averages with Python.* en. July 2020. URL: https://towardsdatascience.com/moving-averages-in-python-16170e20f6c (visited on 07/27/2021).

[24] *NA5W-K RELAY Datasheet pdf - MINIATURE RELAY. Equivalent, Catalog.* URL: https://datasheetspdf.com/pdf/1459339/Fujitsu/NA5W-K/1 (visited on 08/21/2021).

[25] *Predictive Maintenance – Grundvoraussetzungen, Schritte und Möglichkeiten.* de-DE. URL: https://www.pfi.or.at/blog/predictive-maintenance-grundvoraussetzungen-schritte-und-moeglichkeiten-aus-data-analytics-sicht/ (visited on 07/04/2021).

[26] *Productdetails.* en. URL: https://www.omicronenergy.com/de/produkte/cmc-430/ (visited on 08/21/2021).

[27] *Productdetails.* en. URL: https://www.omicronenergy.com/de/produkte/arco-400/ (visited on 08/21/2021).

[28] *Prognostics Center of Excellence - Data Repository.* URL: https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/#turbofan (visited on 08/31/2021).

[29] Yongyi Ran et al. "A Survey of Predictive Maintenance: Systems, Purposes and Approaches". en. In: *arXiv:1912.07383 [cs, eess]* (Dec. 2019). arXiv: 1912.07383. URL: http://arxiv.org/abs/1912.07383 (visited on 07/27/2021).

[30] *RELAY BASICS (3).pdf.* URL: https://www.is-rayfast.com/media/pdfs/lib/Rayfast%20Literature/14%20Relays%20&%20Contactors/RELAY%20BASICS%20(3).pdf (visited on 08/21/2021).

[31] Lotfi Saidi et al. “Wind turbine high-speed shaft bearings health prognosis through a spectral Kurtosis-derived indices and SVR”. en. In: *Applied Acoustics* 120 (May 2017), pp. 1–8. ISSN: 0003682X. DOI: 10.1016/j.apacoust.2017.01.005. URL: https://linkinghub.elsevier.com/retrieve/pii/S0003682X17300130 (visited on 07/22/2021).

[32] Michael Schenk. *Instandhaltung technischer Systeme: Methoden und Werkzeuge zur Gewährleistung eines sicheren und wirtschaftlichen Anlagenbetriebs*. de. Google-Books-ID: IRIjBAAAQBAJ. Springer-Verlag, Oct. 2009. ISBN: 978-3-642-03949-2.

[33] *scipy.stats.iqr — SciPy v1.7.0 Manual*. URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.iqr.html (visited on 07/27/2021).

[34] *scipy.stats.zscore — SciPy v1.7.1 Manual*. URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.zscore.html (visited on 08/23/2021).

[35] Pavel Senin. “Dynamic Time Warping Algorithm Review”. en. In: (), p. 23.

[36] *Short-time Fourier transform*. en. Page Version ID: 1035110534. July 2021. URL: https://en.wikipedia.org/w/index.php?title=Short-time_Fourier_transform&oldid=1035110534 (visited on 07/27/2021).

[37] Chathurangi Shyalika. *Dynamic Time Warping (DTW)*. en. May 2019. URL: https://medium.datadriveninvestor.com/dynamic-time-warping-dtw-d51d1a1e4afc (visited on 07/18/2021).

[38] *sklearn.cluster.KMeans — scikit-learn 0.24.2 documentation*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html (visited on 07/27/2021).

[39] *sklearn.preprocessing.MinMaxScaler — scikit-learn 0.24.2 documentation*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html (visited on 08/21/2021).

[40] *sklearn.preprocessing.RobustScaler — scikit-learn 0.24.2 documentation*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.RobustScaler.html (visited on 08/23/2021).

[41] *Three Ways to Estimate Remaining Useful Life for Predictive Maintenance*. en. URL: https://de.mathworks.com/company/newsletters/articles/three-ways-to-estimate-remaining-useful-life-for-predictive-maintenance.html (visited on 07/04/2021).

[42] Duc Tran Anh, Karol Dąbrowski, and Katarzyna Skrzypek. “The Predictive Maintenance Concept in the Maintenance Department of the “Industry 4.0” Production Enterprise”. en. In: *Foundations of Management* 10.1 (Dec. 2018), pp. 283–292. ISSN: 2300-5661. DOI: 10.2478/fman-2018-0022. URL: https://www.sciendo.com/article/10.2478/fman-2018-0022 (visited on 08/21/2021).

[43] *Understanding LSTM Networks – colah’s blog*. URL: https://colah.github.io/posts/2015-08-Understanding-LSTMs/ (visited on 07/23/2021).

[44] *Wavelet transform*. en. Page Version ID: 1031427861. July 2021. URL: https://en.wikipedia.org/w/index.php?title=Wavelet_transform&oldid=1031427861 (visited on 07/27/2021).

[45] *Wind Turbine High-Speed Bearing Prognosis - MATLAB & Simulink - MathWorks Deutschland*. URL: https://de.mathworks.com/help/predmaint/ug/wind-turbine-high-speed-bearing-prognosis.html (visited on 07/17/2021).

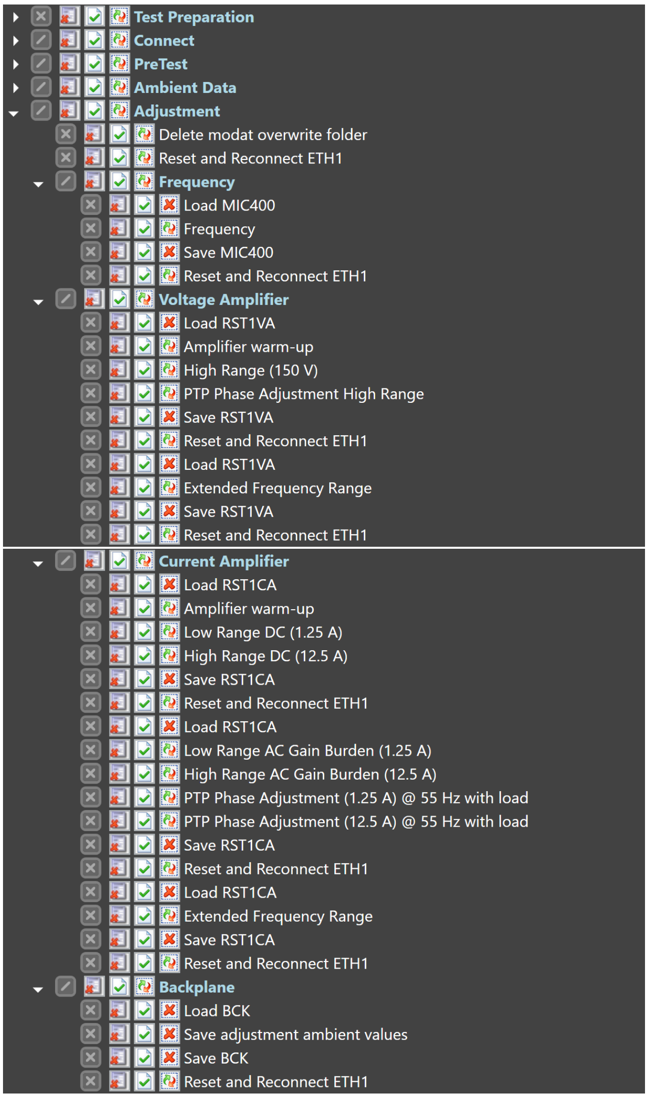
[46] Yumpu.com. *Telecom and Signal Relays - Darlas*. en. URL: https://www.yumpu.com/en/document/view/43212830/telecom-and-signal-relays-darlas (visited on 08/21/2021).

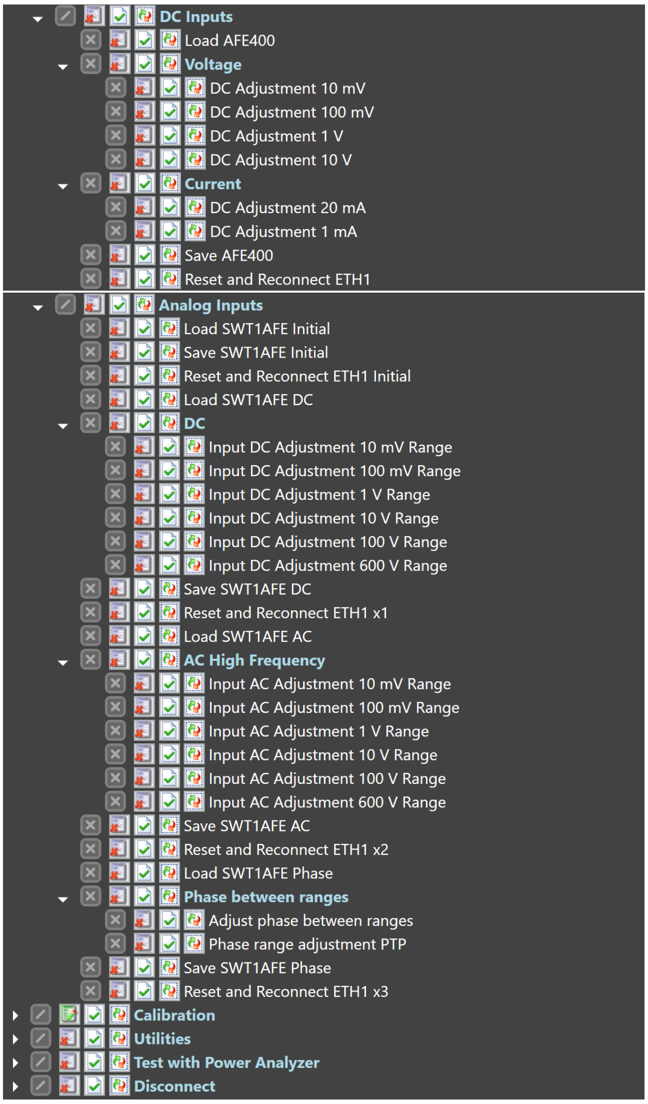
[47] Jeremy Zhang. *Dynamic Time Warping*. en. Feb. 2020. URL: https://towardsdatascience.com/dynamic-time-warping-3933f25fcddd (visited on 07/27/2021).

Appendix

Subtest hierarchy of the CMC430

The images below show the hierarchical structure of the CMC430 unit test, specifically the adjustment part.





9.1 Source code

9.1.1 Database access

```
select
tblTestResult.Serialnumber as serialnumber,
tblReportTestRunReport.TowerName as towerName,
_set.NodeName as _set_nodeName,
_index.NodeName as _index_nodeName,
case
    when ISJSON(tblReportTestRunReport.Equipments) = 1 and json_value(
    tblReportTestRunReport.Equipments, '$[6].Description') = 'ETT1-ISHT'
    then json_value(tblReportTestRunReport.Equipments, '$[6].ToolID')
```

```
    when ISJSON(tblReportTestRunReport.Equipments) = 1 and json_value(tblReportTestRunReport.Equipments, '$[7].Description') = 'ETT1-ISHT'
    then json_value(tblReportTestRunReport.Equipments, '$[7].ToolID')
    when ISJSON(tblReportTestRunReport.Equipments) = 1 and json_value(tblReportTestRunReport.Equipments, '$[8].Description') = 'ETT1-ISHT'
    then json_value(tblReportTestRunReport.Equipments, '$[8].ToolID')
    when ISJSON(tblReportTestRunReport.Equipments) = 1 and json_value(tblReportTestRunReport.Equipments, '$[5].Description') = 'ETT1-ISHT'
    then json_value(tblReportTestRunReport.Equipments, '$[5].ToolID')
    else 'fail'
end as ETT1_ISHT,
case
    when json_value(tblComponentInput.ComponentInputValuesSerialized, '$[17].Name') = 'Nominal'
    then json_value(tblComponentInput.ComponentInputValuesSerialized, '$[17].Value')
    else json_value(tblComponentInput.ComponentInputValuesSerialized, '$[0].Value')
end as Nominal,
case
    when json_value(_index.ComponentResultValuesSerialized, '$[3].Name') = 'Error'
    then json_value(_index.ComponentResultValuesSerialized, '$[3].Value')
    else json_value(_index.ComponentResultValuesSerialized, '$[2].Value')
end as Error,
_index.ExecutionInfo_StartedAt as StartedAt

from tblTestResult
inner join tblTestFinalizationInfo on (tblTestResult.FinalizationInfo_Id = tblTestFinalizationInfo.Id)
inner join tblTestRun on (tblTestRun.TestResult_Id = tblTestResult.Id)
inner join tblComponentResult _set on (_set.TestRun_Id = tblTestRun.Id)
inner join tblComponentResult _index on (_index.ParentComponentResult_Id = _set.Id)
inner join tblReportTestReport on (tblReportTestReport.SequentialTestId = tblTestResult.SequentialTestId)
inner join tblReportTestRunReport on (tblReportTestRunReport.TestReport_Id = tblReportTestReport.Id and tblReportTestRunReport.Position = tblTestRun.Position)
inner join tblReportTestRunItem on (tblReportTestRunItem.TestRunReport_Id = tblReportTestRunReport.Id)
inner join tblTestInput on (tblTestInput.Id = tblTestResult.TestInput_Id)
inner join tblComponentInput on (tblComponentInput.TestInput_Id = tblTestInput.Id and tblComponentInput.StaticComponentId = _index.StaticComponentId)

where (tblTestResult.DeviceType='CMC430')
and (_index.NodeName='0 V Adjustment')                                          -- condition to select node
-- and (_set.NodeName='VL1')
and (tblReportTestRunItem.Header='Adjustment  \  Voltage Amplifier  \  High Range (150 V)')      -- condition to select subtest
```

```
and(tblReportTestRunReport.TowerName = 'Epsilon01' or
    tblReportTestRunReport.TowerName = 'Epsilon02' or
    tblReportTestRunReport.TowerName is null)
and len(_index.ComponentResultValuesSerialized) < 1000      -- getting rid of json with subtest informations
and (_index.ComponentResultValuesSerialized <> '[]')
and (_index.ComponentResultValuesSerialized <> '[]')
and (tblReportTestRunReport.Equipments <> '[]')
and (tblReportTestRunReport.Equipments <> 'null')
and len(tblReportTestRunReport.Equipments) > 1000
order by _index.ExecutionInfo_StartedAt
```

Listing 9.1: Fetching data for subtest - example

9.1.2 Trendability analysis

```
import numpy as np

def trendability(dataset, feature, time_is_index_values=False):
    feat = feature
    t = None
    if time_is_index_values:
        t = (dataset.index.to_series() - dataset.index.to_series()[0])  # series of operation time in numOfIndexValues
    else:
        t = (dataset.index.to_series() - dataset.index.to_series()[0]).dt.total_seconds() / 86400    # series of operation time in days
    a = (dataset[feat] * t).sum()
    b = dataset[feat].sum() * t.sum()
    c = (dataset[feat]**2).sum()
    d = dataset[feat].sum()**2
    e = (t**2).sum()
    f = t.sum()**2

    # calculate trendability R
    n = len(dataset)   # number of measurements
    R = (n*a - b) / (np.sqrt((n*c - d) * (n*e - f)))
    return R
```

Listing 9.2: Algorithm for calculating the trendability

9.1.3 Monotonicity analysis

```
def monotonicity(dataset, feature):
pos_derivs, neg_derivs = [], []
feature = feature + 'l1'
# print(feature)
for el in dataset[feature]:
    if el > 0:
        pos_derivs.append(el)
```

```
        elif el < 0:
            neg_derivs.append(el)
    return abs(len(pos_derivs) - len(neg_derivs)) / (len(dataset) - 1)
```

Listing 9.3: Algorithm for calculating the monotonicity

Statement of Affirmation

I hereby declare that all parts of this thesis were exclusively prepared by me, without using resources other than those stated above. The thoughts taken directly or indirectly from external sources are appropriately annotated. This thesis or parts of it were not previously submitted to any other academic institution and have not yet been published.

Dornbirn, Gabdelgaziz Sayfutdinov