David Reiter, BSc

Development of Concepts for Remaining Useful Lifetime Estimation of Components in Railway Application

Master's Thesis

to achieve the university degree of

Master of Science

Master's degree programme: Mechanical Engineering

submitted to

Graz University of Technology

Supervisor

Ass.Prof. Dipl.-Ing. Dr.techn. Christian Moser

Institute of Machine Components and Methods of Development

Second Supervisor

Assoc.Prof. Dipl-Ing. Dr.mont. Franz Pernkopf

Signal Processing and Speech Communication Laboratory

Graz, June 2018

Affidavit

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master's thesis.

Date (D. Reiter)

Abstract

Nowadays competitive products within the railway industry have to offer more than the basic functionality, which is why the future trend goes towards the digital train. That is a revolutionary approach, that aims to improve railway mobility simultaneously at different levels of product development, using intelligent assistance systems. Condition based maintenance is identified as one of the most promising applications of such systems, which require condition detection and prediction tools to assess the vehicles components.
This thesis deals with the development of an algorithm, that estimates the remaining useful lifetime (RUL) of bogie components, with focus on data driven concepts for dampers. A so called multi-branch hidden Markov model (MB-HMM) is implemented, to represent the evolution of a components health condition over time, for diverse, non-linear deterioration modes. Deficiencies regarding restrictions of its input data are compensated by a previously performing classification using support vector machines (SVM), that enables processing of multi-dimensional data sets. The necessary data for machine learning of statistical models is simulated by a linearized, three-dimensional multi-body system of a four-axle railway vehicle.
After the successful learning of SVM and MB-HMM for rapid, average and slow deteriorating dampers, the models are tested with independent data. An iterative procedure provides successively increasing segments of the simulated data, which enables an assessment of the model's accuracy. The algorithm estimates the RUL of average and slowly degrading dampers, end-to-end extraordinary well, whereas the rapid deterioration requires 75% of the overall data, in order to produce an acceptable output.
Several extensions offer possibilities for optimization, primarily in terms of a more universal form of permitted input data, which can be used to improve the model accuracy for all branches. Thereby the MB-HMM can be used for a RUL estimation of different components, provided that the data situation allows machine learning.

Kurzfassung

Heutzutage müssen wettbewerbsfähige Produkte im Bereich der Schienenfahrzeug Industrie mehr als die Basisfunktionalität bieten, weshalb die Zukunft in Richtung des digitalen Fahrzeuges geht. Das ist ein revolutionärer Ansatz, der versucht mittels intelligenter Assistenzsysteme gleichzeitig mehrere Ebenen der Produktentwicklung zu verbessern. Als eine der vielversprechendsten Anwendungen solcher Systeme wird die zustandsbasierte Instandhaltung identifiziert, welche sinngemäß nur mit erfolgreicher Zustands- und Restlebensdauerbestimmung (RUL) der Fahrzeugkomponenten funktioniert. Diese Arbeit beschäftigt sich mit der Entwicklung eines Algorithmus zur Bestimmung der Restlebensdauer von Fahrwerkskomponenten, wobei der Schwerpunkt auf datengetriebenen Konzepten für Dämpferelemente liegt. Eine zeitliche Auflösung der Entwicklung des Zustandes einer Komponente für unterschiedliche, nicht lineare Verschleißmechanismen, wird mit einem sogenannten Multi-Branch Hidden Markov Modell (MB-HMM) abgebildet. Defizite des Modells hinsichtlich dessen Restriktionen der Eingangsdaten werden durch einen vorgeschalteten Klassifizierer in Form einer Support Vector Machine (SVM) kompensiert, die es ermöglicht mehrdimensionale Datensätze zu verarbeiten. Die notwendigen Daten für das maschinelle Lernen der statistischen Modelle werden mit einem linearisierten, dreidimensionalen Mehrkörpersystem eines vierachsigen Schienenfahrzeuges simuliert. Nach erfolgreichem Lernen von SVM und MB-HMM für schnell, durchschnittlich und langsam verschleißende Dämpfer, werden die Modelle mit unabhängigen Daten getestet. Ein iteratives Verfahren, welches den Modellen schrittweise wachsende Anteile dieser simulierten Daten zur Verfügung stellt, ermöglicht eine Bewertung deren Genauigkeit. Für durchschnittlich und langsam degradierende Dämpfer schätzt der Algorithmus die RUL durchgehend außerordentlich genau, wohingegen schnelle Abnutzung erst nach 75% der verstrichenen Gesamtlebensdauer akzeptabel bestimmt werden kann. Zahlreiche Erweiterungen, die sich auf eine universellere Form der möglichen

Eingangsdaten beziehen, bieten Raum zur Optimierung, welche die Modellgenauigkeit aller Branches erhöht. Damit kann das MB-HMM für RUL Bewertungen verschiedener Komponenten herangezogen werden, vorausgesetzt die Datenlage erlaubt maschinelles Lernen.

Contents

Abstract iii

Abbreviations x

1. Introduction 1
- 1.1. Motivation for Prognostic Concepts . . . 1
- 1.2. Maintenance Strategies . . . 3

2. Predictive Maintenance and its Impact 7
- 2.1. Comparison of Applied Maintenance Strategies . . . 8
- 2.2. A Survey of Life Cycle Costs for Railway Vehicles . . . 9

3. Fundamentals of Data Driven Prognostics 12
- 3.1. Data Driven Prognosis . . . 14
- 3.2. The Machine Learning Diagram . . . 16
- 3.3. Training Data for Mechanical Dampers . . . 18
  - 3.3.1. Acquisition of Training Data . . . 18
  - 3.3.2. Feature Selection . . . 21
- 3.4. The Course of Action . . . 22

4. Implemented Methods 24
- 4.1. A Linear Multi-Body Simulation . . . 25
  - 4.1.1. Data Acquisition Using a Multi-Body Simulation . . . 26
  - 4.1.2. Model of a Railway Vehicle . . . 27
  - 4.1.3. The Newton-Euler Procedure . . . 31
  - 4.1.4. Numerical Solution of the Equation System . . . 47
- 4.2. Classification with Support Vector Machines . . . 50
  - 4.2.1. Learning of a Support Vector Machine . . . 51
  - 4.2.2. Testing a Support Vector Machine . . . 56
  - 4.2.3. Length and Composition of the Synthetic Sequence . . . 57

4.3. Prognostics with Hidden Markov Models 60
4.3.1. The Three Problems of Hidden Markov Models 65
4.3.2. Extension of the Left-Right Hidden Markov Model . . . 76
4.3.3. Solution for the Remaining Useful Lifetime Algorithm . 80

5. Results **87**
5.1. The Equation of Motion by a Multi-Body Simulation 88
5.2. Mapping of the Health Condition onto Acceleration Signals by Support Vector Machines 91
5.3. Prediction of the Remaining Useful Lifetime by Multi-Branch Hidden Markov Models 94

6. Conclusion and Prospect **101**

Appendix **105**

A. The State Space Model **106**
A.1. An Example of the Newton Euler Procedure 106
A.2. Details to the Connection Equations 108

B. Pseudo Code **111**

Bibliography **116**

List of Figures

1.1. Tradeoff between investment costs (preventive costs) and maintenance costs 5

2.1. Bogie components 7
2.2. Benefit of condition based maintenance (CBM) 9
2.3. Life cycle cost model [12] 10
2.4. Relative share of life cycle costs 11

3.1. Classification of machine learning models 14
3.2. The machine learning diagram [1] 17
3.3. Acquiring training data 19
3.4. The course of action to implement a remaining useful lifetime (RUL) algorithm 22

4.1. A railway vehicle represented by a multi-body system. 28
4.2. Non-linear wheelprofile. 36
4.3. Principle of track guiding. 37
4.4. Wheelset - effective forces. 40
4.5. Bogie - effective forces. 44
4.6. Carbody - effective forces. 46
4.7. Maximum margin hyperplane for a two-class support vector machine (SVM). 52
4.8. Kernel transformation from input space X to feature space F. . 55
4.9. Fit of a posterior probability to the score. 56
4.10. Poisson distribution of health state evolution 58
4.11. Normal distributed maximum mileage of a damper 60
4.12. hidden Markov model (HMM) Automaton. 62
4.13. HMM Sequence of states and observations. 63
4.14. Possible state transitions for different types of HMMs. 64
4.15. State space trellis of states for a given observation sequence. . . 67

4.16. Computation scheme for recursion of forward variable $\alpha_{t+1}(i)$. 68
4.17. Computation scheme for recursion of backward variable $\beta_t(i)$. 73
4.18. Computation scheme for joint probability $\zeta_t(i,j)$. 74
4.19. Multiple branch, left-right automaton for modeling of coexistent failure modes. 78
4.20. Framework for RUL estimation with multi-branch hidden Markov model (MB-HMM). 81
4.21. PDF for different states (rows) within the emissions matrix $\boldsymbol{B}$. . 83

5.1. Transfer function of the vertical positions from car body and track excitation 89
5.2. Curvature and corresponding lateral system response 90
5.3. Hyperplane of a two class SVM 92
5.4. Score probability mass function (PMF) of the SVM 93
5.5. Posterior probability computed by the selected SVM. 94
5.6. Sequence of states for three branches. 96
5.7. Sequence of states for testing data and training data. 97
5.8. RUL estimation for three different testing sequences. 99

A.1. Simplified vehicle model. 107

Abbreviations

CBM condition based maintenance
CoG center of gravity
DoF degrees of freedom
EoL end of life
EoM equation of motion
FIM fixed interval maintenance
HMM hidden Markov model
HS health state
HSMM hidden semi Markov model
IoT internet of things
LCC life cycle costs
MB-HMM multi-branch hidden Markov model
PM predictive maintenance
PMF probability mass function
RTFM run to failure maintenance
RUL remaining useful lifetime
SV support vector
SVM support vector machine
TBM time based maintenance

1. Introduction

It does not matter whether transportation goes through the air, on water or on the ground. The need of humanity to travel from A to B is relentlessly fueling the research and development for better and cheaper technical solutions in all branches. From customer perspective, costs, safety, comfort and time, as defined by punctuality and travel duration are the main aspects which affect the decision between means of transportation. The improvement of those four properties can be seen as the superior goal for the contractor.
Needless to say the real world is not as black and white, which probably makes the above statement seem naive. However, regardless of social, economical or ecological boundaries, which may as well effect a customers decision, the above mentioned properties serve as superior development requirement. Improving any of them is going to help win the competition within and outside of the branch. The following section discusses which development requirements can be enhanced by prognostic concepts.

1.1. Motivation for Prognostic Concepts

Given that the history of railway industry goes back to the early 19th century, railway vehicles as we know them have been developed for a rather long time. Despite the fact that the variety of types, in terms of system architecture and engine concepts has been growing vast, railway vehicles can still be considered as well-proven and reliable. Thanks to knowledge and experience built throughout the course of history, quality meets a high standard. Nevertheless room for improvement has to be identified in order to stay competitive in the open market.

Similar to the industrial revolution at the beginning of the 19th century, the digital revolution has been pushing technical progress ever since the end of

the 20th century. That is how a basically purely mechanical system like a railway vehicle, becomes upgraded by digitization. A once "disconnected" stand alone vehicle, gets equipped with sensor systems, processing units and a network interface. Thereafter able to observe, transmit and share data of itself, the vehicle is no longer on its own, but "connected" to a network. This "digital train" is linked with a virtual world, exchanging information at any given time. From a general point of view, "connecting" all objects to a global network is a vision which is called the internet of things (IoT). A vision, which has become partially real over the course of the last decade. Its purpose is to interconnect all objects, giving them the opportunity to interact intelligently [22]. Within the IoT all objects are capable of observing, transmitting and processing data in a way, that optimizes their utility for the user. The implementation of such an intelligence requires a huge amount of data as well as algorithms, which are able to interpret it and autonomously set necessary measures. Those algorithms should recognize patterns, classify data and predict upcoming events. Depending on the use case, the IoT detects different quantities and derives different measures. Applications range from a supply chain management in the pharmaceutical industry, to smart machines with the purpose of an increased availability. In the railway industry, digitization grants access to time correlated, health related data and thereby enables condition monitoring of system components as well as an improved development process, due to a better exploitation of resources.

The IoT whithin the railway industry, aims to interconnect vehicles, tracks and related institutions, which all are equipped with countless electronic devices [5]. They observe data about their health state and share information about their utilized capacity. By processing these observations appropriately, availability[1] can be optimized, which keeps the downtime to a minimum. This is exactly the purpose of a health monitoring system used for condition based maintenance (CBM) and predictive maintenance (PM). Both of them seem to be effective strategies in comparison with time based maintenance (TBM), where a fixed time or mileage limit[2] determins the next maintenance stop. The objective of CBM and PM is to maintain a vehicle only when its necessary [4]. Taking into account that besides an optimized maintenance plan,

[1] Availability is the ratio between the time a system is working as it is supposed to and the overall time of operation [4]

[2] The minimum limits are defined by national regulations and laws [4]

those strategies are capable of minimizing safety-endangering incidents and improving the development process, their benefit is truly huge. The latter can be achieved by rationalizing springs and dampers, or optimizing the design of structure parts with respect to weight. The biggest benefit however, is the improved maintenance strategy, which will be argued in section 2.2.

To successfully run such a maintenance strategy, certain algorithms for failure detection and prediction are required. Their purpose is to convert the observed sensor data to characteristic values that represent the components health state and its future evolution. Concepts for diagnostics and prognostics are manifold and need to be investigated further to apply them appropriately. Especially prognostic concepts require increased attention in order to develop their full potential.

1.2. Maintenance Strategies

The maintenance strategy is a holistic guideline for the management of maintenance related resources[3]. Its output is a maintenance process, which ideally does not leave any related actions undefined. Clearly the nature of this strategy has a major impact on the required financial funds and on the downtime of the maintained vehicle. During the last decade the maximization of the availability became an important topic. The concept of CBM and PM emerged, where diagnostic and prognostic concepts identify and predict incipient failures [18].

Less complex approaches are run to failure maintenance (RTFM) or fixed interval maintenance (FIM) as shown in table 1.1. Assuming a component would brake down during operation, RTFM provides only corrective actions, no matter if the failure results in any safety or economically critical event. FIM on the other hand, uses corrective actions only for non-critical failures. Critical failures are tackled by preventive actions, which are performed during fixed maintenance intervals. During the span of time between those depot checks however, failures can not be detected. Therefore, the frequency of intervals determines the probability of detecting a failure. Usually a lot of experience and expert knowledge is used to define this frequency, as well as quantities

[3]Those contain all involved materials, tools and employees

	Strategy		
	Run to Failure Maintenance (RTFM)	**Fixed Interval Maintenance (FIM)**	**Condition Based & Predictive Maintenance (CBM & PM)**
Corrective Action	General concept for RTFM	Non-critical failures	Non-critical failures
Preventive Action	Not included	Critical failures. Fixed interval	Critical failures. Dynamic interval after detection.
Predictive Action	Not included	Not included	Critical failures. Dynamic interval based on prediction.

Table 1.1.: Maintenance strategies and their actions [18]

with limit values for all relevant components. Evaluated in course of the depot check, those quantities indicate the necessity of any maintenance measure. RTFM and FIM both are "Time Based Maintenance" (TBM) strategies. Maintenance actions are performed after a certain elapsed time or driven mileage, regardless of the considered components health status. Easy to see that TBM has a notable risk of either not performing if necessary, or the other way round. Components could be removed way before their end of life (EoL), or already been broken but still in operation, decreasing safety and harming other components. Both ways, unnecessary costs are provoked, which is why CBM and PM are logical successor strategies. Figure 1.1 shows qualitative the tradeoff between preventive costs and actual maintenance costs. Preventive costs include everything from the sensor hardware of the monitoring system, to the front end software. Maintenance costs are considered to increase, if unscheduled downtime occurs. The total costs, are the sum of both of them. RTFM does not have a lot of preventive costs, however huge maintenance costs due to the certainly occurring unscheduled downtime. FIM is considered to have rather short maintenance intervals, to guarantee operation with very less unscheduled downtime. CBM aims to find the optimum between investment costs (preventive costs) and maintenance costs. CBM[4] is a completely differ-

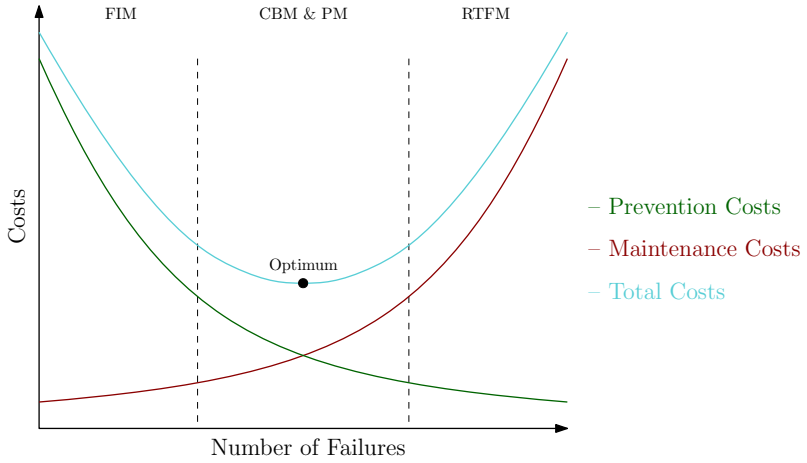


Figure 1.1.: Tradeoff between investment costs (preventive costs) and maintenance costs

ent approach. Whereas TBM reacts only after a failure already occurred and supports discrete observation, CBM predicts when maintenance action will be necessary by observing the components health continuously. The latter implies that the vehicle is equipped with a monitoring system. That is a set of sensors which record the dynamic system response of the vehicle during operation. This recorded data is used as input for algorithms, which are responsible for failure detection and prediction of RUL. In a manner of speaking, CBM is one step ahead of TBM.
As table 1.1 points out, preventive action is performed immediately when detecting a failure. The fixed maintenance interval from TBM becomes dynamic, because observations are possible at any given time. On top of that, predictions of the components future health enable an optimized scheduling. With that information, the possibility of a broken component in operation is minimized, thus safety is increased and unscheduled downtime is lowered. The necessary maintenance actions and their occurrences are also predictable, which makes resource planning easy. All those benefits can only be realized with sophisticated diagnostic and prognostic concepts. Based on continuously observed sensor data, those concepts determine the current health state and the RUL, which is basically the evolution of the health state from the current

[4]from here on out CBM and PM will be referred to as CBM

state until the EoL. CBM has advantages that can contribute to the major development requirements from chapter 1 as follows:

- Costs
  - Optimized maintenance scheduling
  - Optimized resource management
  - Minimized unscheduled downtime
  - Utilization until close to EoL
- Safety
  - No broken components in operation
- Time - punctuality
  - Minimized unscheduled downtime

In order to give a concrete example of what can be accomplished by an advanced maintenance strategy, chapter 2 compares the currently used process with a condition based approach at SIEMENS.

2. Predictive Maintenance and its Impact

A railway vehicle is a very complex system with high demands on safety and availability. Typically the vehicle is composed of several wagons, which are in turn composed of two bogies and a car body. The bogie is indeed one of the most important components. It is responsible for carrying the car body, track guiding and transmission of driving as well as brake power [9]. The main sub-components are illustrated in figure 2.1. Railway vehicles are designed to

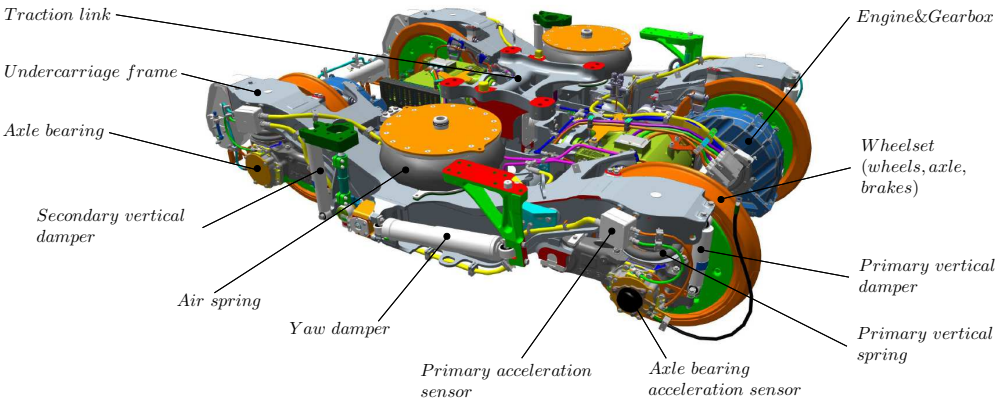


Figure 2.1.: Bogie components

be in operation for up to thirty years. A rather long time to run, for highly stressed mechanical parts. In order to keep their quality and functionality, they need to be maintained properly.

2.1. Comparison of Applied Maintenance Strategies

SIEMENS currently pursues a TBM strategy. There is no such thing as a monitoring system, hence observations of the components health states are rare. In order to guarantee a certain level of safety and comfort, maintenance inspections have to happen pretty often. A maintenance plan defines how often those inspections take place and which dimensions have to be checked in order to decide if a component has to be replaced or repaired. In case of reparation, the plan defines all necessary maintenance actions. However, if the reviewed dimension does not exceed its predefined limits, it is going to be in operation until the next inspection. Anything that happens in between will be unnoticed. Maintenance intervals are organized as follows.

- Visual inspection
- Damper replacement
- Brake revision
- Complete bogie revision
- Axle bearing replacement

All of them have to be done at latest before the vehicle reaches a specified mileage. Considering that, minimizing downtime is a real challenge and not without risk. Those checks are scheduled in a way, that as many tasks as possible can be done within one inspection. Therefore some of them are performed early and some are rescheduled.

With a CBM strategy, being able to predict the components RUL, scheduling is much easier. Downtime can be calculated in advance, because continuously observed health conditions allow an early decision of required maintenance actions. Necessary resources can be organized beforehand, which can be a major impact, considering the long lead times of certain components or the availability of special maintenance tools[1]. However, the regularity of the above mentioned maintenance checks can not be completely overruled by CBM. A health monitoring system can also be faulty and there may exist some fault modes that can not be detected by the algorithms properly. Therefore the existing maintenance intervals are not canceled, but extended by a certain mileage. Figure 2.2 shows an estimate of the expected benefit for a typical

[1] Underfloor lathe

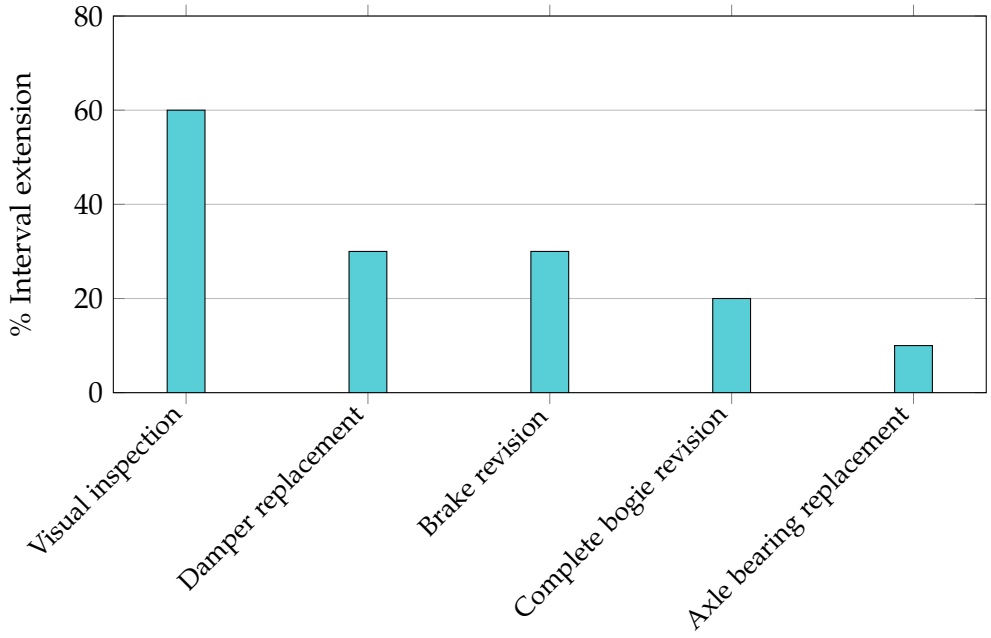


Figure 2.2.: Benefit of CBM

commuter train. The maximum mileage, that defines the window within which maintenance is mandatory, can be increased by up to 60%, depending on the type of inspection.

Since this leads to a cost reduction, the following section roughly breaks down a railway vehicles' life cycle costs, in order to understand the impact of a well designed CBM system, which is based on reliable diagnostic and prognostic algorithms.

2.2. A Survey of Life Cycle Costs for Railway Vehicles

Estimating the total costs of ownership for a product or a system is a question of boundaries. The resulting estimation depends on whether certain aspects are included or excluded. Taking into account that currently developed railway

vehicles are designed to be turnkey solutions, life cycle costing[2] has to consider not only the vehicle itself, but also the entire railway system. Therefore design, development, production, installation, operation, maintenance, support and disposal of vehicle and infrastructure are included into the consideration. An approach like this, leads to quite accurate estimations of the life cycle costs (LCC), which corresponds very well to the total costs of ownership [12].

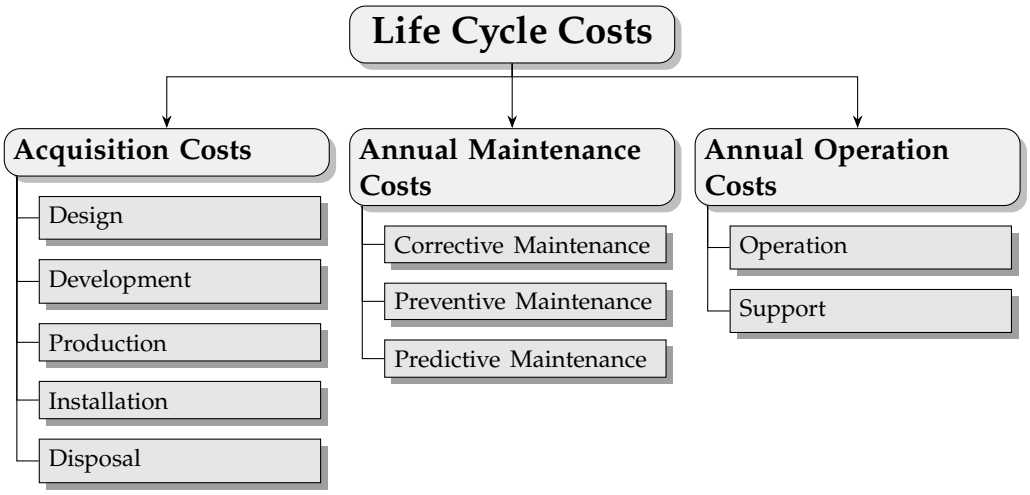


Figure 2.3.: Life cycle cost model [12]

As shown in figure 2.3, those aspects can be roughly categorized into acquisition, maintenance and operation. Acquisition costs occur only once, which explains why disposal is a part of it. The other two aspects are annually recurring. In which way the maintenance costs are divided, is obviously dictated by the maintenance strategy.
A quantification and qualification of the LCC is a rather complex task, which is beyond the scope of this thesis. However, the public literature offers several different approaches, where the LCC of every sub-component is added up to get a result for the entire railway vehicle. An absolute value as well as a relative share of each cost aspect is the desired result of life cycle costing. A huge number of variables can alter this result, which is why this task is anything but trivial. Nevertheless, a very rough qualification should point out

[2]Process of cost estimation throughout the whole life cycle [12]

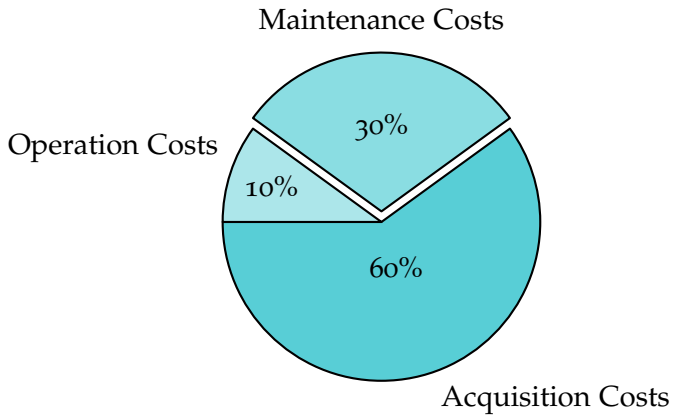


Figure 2.4.: Relative share of life cycle costs

the potential of a sophisticated CBM system. Figure 2.4 is an estimation of the relative share of LCC, whereby CBM is not established. It shows that an improved maintenance system can contribute a great deal to cost optimization. As mentioned above, maintenance costs recur annually, causing a major part of the absolute LCC. Considering the maintenance interval extensions from section 2.1, which are 30% on average, a maintenance cost reduction from 30 to 20% is possible.

In conclusion, prognostic concepts are a necessary feature for CBM systems, which have a great potential in terms of decreasing costs and increasing safety. The development of reliable prognostic concepts is topic of this thesis and will be presented in detail.

3. Fundamentals of Data Driven Prognostics

Estimating the RUL for mechanical components in railway vehicles is an essential part of the CBM strategy. Any component's current condition, by means of faulty or healthy, can be described by its health state (HS). Hence the future condition can be evaluated by the HSs evolution over time. In this context, RUL is defined by the current HS and the time or mileage until it exceeds a reasonable threshold. An algorithm that computes the RUL has two tasks:

1. Determine the current health state (detection)
2. Estimate its future evolution (prediction)

Those two tasks are basically executed recurrent, in daily or weekly intervals. Therefore, every time the algorithm predicts the components RUL, additional data, observed since the last interval, is available. Hence, in course of the components life, the RUL estimate will increase in accuracy, because more data is provided. To assess the algorithm during its development, this procedure is artificially recreated. Details of the framework are provided in section 3.4.

There are many possibilities for developing such an algorithm. The key is to figure out what the advantages and disadvantages of the different methods are and how they apply to the particular situation or problem. All those methods seek to establish a model that approximates the reality as good as possible and ultimately deliver a statement for the RUL. There is however, a difference in how those models are created, which leads to a distinction as shown in table 3.1. The basic idea of both, physical and empirical models, is to find a quantity that correlates to the HS of a component and predict its evolution over time. This quantity has to change with ongoing deterioration in order to make an estimation possible. A physical model works with mathematical formulations

	Physical Model	**Empirical Model**
Source	Physical laws	Observed data
Implementation	Mathematical formulation	Machine learning
Approach	Analytic	Numeric
Advantage	No measurements required. Application in an early development stage.	Can represent transient and complex systems
Disadvantage	Oversimplification	Greatly dependent on data set

Table 3.1.: Prognosis Models

of physical laws or hypotheses, that approximate a certain failure mode. This reduces their application to components, whose relevant failure modes are known physical relations. If those relations are hypotheses instead of laws, a bigger simplification error has to be considered. On the other hand, their implementation is basically straight forward, which allows them to be used in an early development stage. In contrast, an empirical model[1] works with observed data, which enables the representation of transient and complex relations. Usually empirical models require an independent data set from which the target relation can be derived. This process is called learning or training and is performed by a certain algorithm. Generally speaking, learning a relationship between certain quantities to classify data is called "machine learning". Because of its independence of knowledge about any physical relations of failure modes and its possibility to represent a target function of any complexity, the data driven approach is chose to be implemented.

Figure 2.1 shows the main mechanical components of a bogie. Certainly all of them would have to be monitored and targeted from the detection and prediction algorithms. The fact that the components deterioration characteristics are very different from each other, supports the decision to use a data driven model for the prediction algorithm. However, not only their deterioration, but also their relevant quantities that indicate any change of health condition, are very different. Since the approach of machine learning is the same, but the pre-processing and data acquisition is not, this thesis focuses only on

[1] Also referred to as data driven model.

prediction of damper elements. Precise it deals with the secondary vertical damper, that connects the car body with the bogie.

3.1. Data Driven Prognosis

The data driven approach can be realized with machine learning algorithms. Machine learning, as a scientific discipline, is a subfield of artificial intelligence. Its objective is to automatically recognize patterns out of data and consequently derive decisions [3]. Models of machine learning are almost as manifold as its applications. In figure 3.1, they are classified by their nature of learning and their field of application.

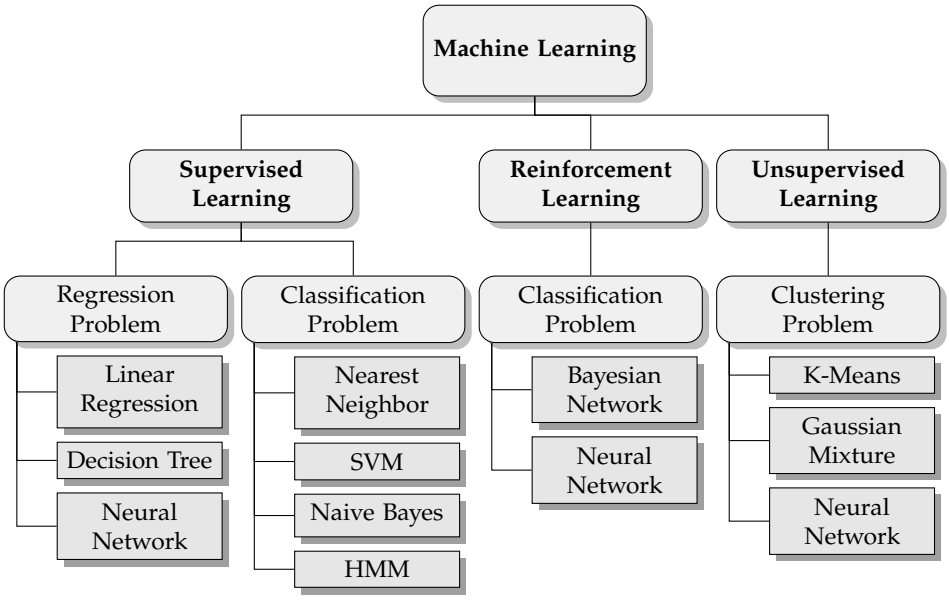


Figure 3.1.: Classification of machine learning models

Unsupervised Learning In this setting, training data does not contain any information about the target quantity. In an engineering application, it consists

of a set of sensor signals or derived feature quantities in the form $\mathcal{D} = \boldsymbol{x_1}, ..., \boldsymbol{x_N}$, where N is the length of the data set [1]. Unsupervised learning is used to solve a clustering problem. Models are able to separate data into certain clusters. However it is not possible to determine the class of the cluster. Certainly they recognize the difference between a healthy and a faulty component, but not which cluster is healthy and which one faulty.

Reinforcement Learning The training data for reinforcement learning does have some information about the target output, which is however, not the target quantity itself. In fact, every observed signal value comes with a possible output and a measure that rates this output, in the form $\mathcal{D} = \boldsymbol{x_1}, \tilde{y}_1, w_1, ..., \boldsymbol{x_N}, \tilde{y}_N, w_N$. It is mainly used to learn how to play a game, whereby every move in a certain stage is rated, which enables the algorithm to figure out the best line of play [1].

Supervised Learning We talk about "supervised", when the training data set consists of the above mentioned feature quantities, as well as a corresponding target value in the form $\mathcal{D} = \boldsymbol{x_1}, y_1, ..., \boldsymbol{x_N}, y_N$. The quantity y_i, $\{i \in \mathbb{Z} | 1 \leq i \leq N\}$ is also called the "label" and can be anything from a binary number like 0 or 1, to a verbal tag like *blue* or *red*. Models of the regression problem differ from those of the classification in their mathematical output space. Whereas regression deals with real output values $y \in \mathbb{R}$, classification is only binary and labels data as 0 or 1, *yes* or *no*, *blue* or *red* or any other class. Since 0 and 1 are also real values, a regression problem can more or less classify as well. The definition is a little bit vague, which is why there may exist different classifications in the literature. Supervised learning is a popular method for condition based prognostics in engineering.

The selection of a suitable model is supported by the technical expertise of the analytics group, which is responsible for bogie diagnostics and prognostics within SIEMENS rolling stock engineering. This knowledge is very welcome, if not required, when it comes to the decision, which actual model is going to be implemented. Since a concrete labeling of the observed data to the HS of the component is required, it is going to be a supervised learning model. Based on expert knowledge and literature research, the implementation of a HMM turned out to be a promising method. For this application, it is a statistical

model, that connects physical states of a component with certain transission probabilities. Those states are basically unknown. They can be estimated by observations, which are emitted by a state with a certain probability. Section 4.3 provides a detailed explanation of HMMs. Its probabilistic connections enable a non-linear propagation of the components condition from new to broken. For the basic HMM however, the observed emissions used in the model, are restrited to be one-dimensional. The health state and its evolution hast to be estimated by only a single quantity, which is certainly not an acceptable way. To tackle that problem, any classifier is necessary before the HMM, to process multi-dimensional input to one-dimensional quantities. Since SVMs are already applied and working well within diagnostics algorithms, they are going to be used. Their outstanding ability of generalizing[2], as well as a nice visual representation of the results for low-dimensional problems, are just two reasons for the selection of this method. Section 4.2 provides more details to SVMs. The combination of a SVM and an HMM have to deal with the two basic requirements for prognostic algorithms, which are the detection of the current health state and the prediction of its future evolution. The presented thesis gives a holistic overview of the workflow and provides details to the implementation of the algorithm.

3.2. The Machine Learning Diagram

SVMs and HMMs are both part of the category supervised learning, which means that they are trained with labeled data. How the process from learning to testing actually works is schematically shown in figure 3.2.

To describe the process of learning and testing, a two-dimensional input space is assumed, where the input quantities are acceleration signals from a bogie and the output is a class, which can either be “healthy” or “faulty”. That would be an example of two class learning. Any machine learning problem starts with a target function $f : \mathcal{X} \rightarrow \mathcal{Y}$. For linearly separable data, this is a straight line, that separates healthy from faulty data points. That target

[2]Generalization is the property of a model, that quantifies the difference between the in sample and the out-of-sample error. In sample error relates to training data, out-of-sample error to test data [1]

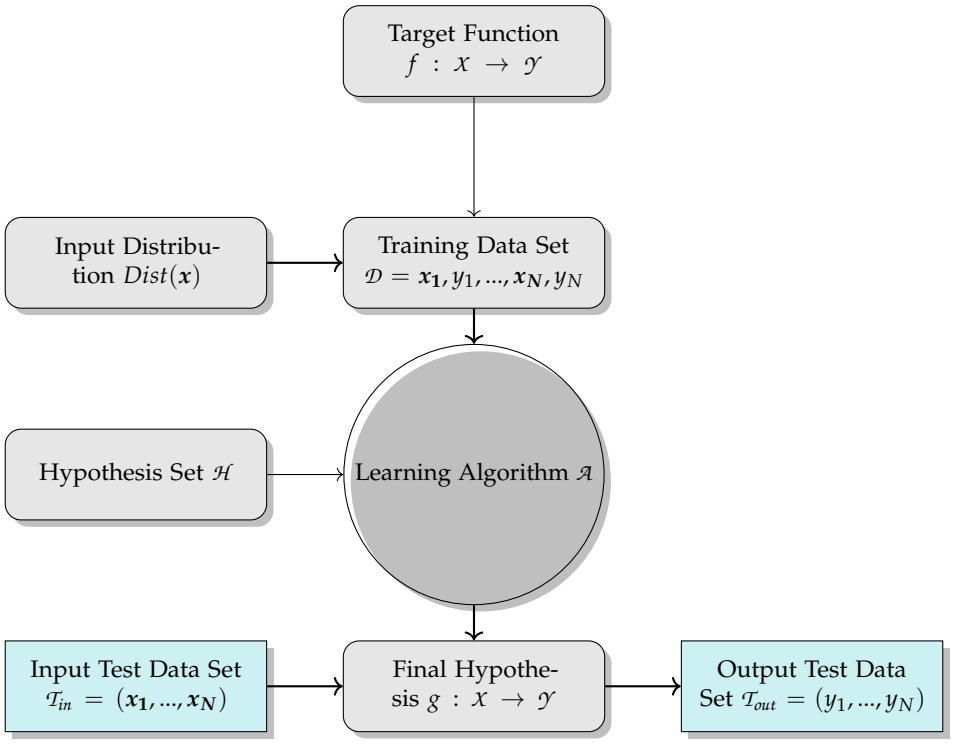


Figure 3.2.: The machine learning diagram [1]

function is unknown and is supposed to be approximated by the machine learning model. To estimate that function, a training data set is required, which provides acceleration signals for healthy and faulty parts. That means that every pair of input signals x_i, $\{i \in \mathbb{N} | 1 \leq i \leq N\}$ is tagged with the label "healthy" or "faulty", so that the algorithm is able to distinguish between them. The learning algorithm is supposed to find the best estimation $g \approx f$ out of the hypothesis set $\mathcal{H}$. In this two-dimensional linear example, $\mathcal{H}$ would consist of an infinite number of straight lines in the input space. Once the learning algorithm picked the final hypothesis, the model is ready for testing. The testing data set consists only of observations. With the learned function, the model can determine whether the test data corresponds to "healthy" or "faulty" parts, by figuring out on which side of the line the test examples are. Hence, a label can be added to the testing data set, giving information about

the current health state of the considered component. The whole process can be divided into

- Training (Learning) and
- Testing

which is basically valid for all different models, hence also for SVMs and HMMs. Since machine learning models are almost exclusively of statistical nature, every statement made by an algorithm comes with a probability. The quality of that statement strongly depends on the training data used to find the final hypothesis. In the above mentioned example, input quantities were assumed as acceleration signals from the bogie. That raises the question, which actual acceleration signals are a good choice. Within machine learning that process is called feature selection. It aims to find the right number and type of input quantities, in order to increase the accuracy of diagnostic and prognostic algorithms. Since the deterioration of every component of a bogie has a different impact on the dynamic behavior of the vehicle, the significant features are different as well. This thesis focuses on the prognosis of dampers in railway vehicles, which is why in the following sections the procedure will be described specifically for this component.

3.3. Training Data for Mechanical Dampers

This section discusses how proper training data can be acquired and which features it should consist of, to enable a RUL prognosis. Basically its form is $\mathcal{D} = \boldsymbol{x_1}, y_1, ..., \boldsymbol{x_N}, y_N$. For an engineering application like the damper, the label y is the components HS. The corresponding observations x are features, extracted from signals of the health monitoring system. These observation and label pairs have to be acquired throughout the dampers whole lifetime. Meaning that there have to be observations with healthy and faulty labels.

3.3.1. Acquisition of Training Data

Just like most mechanical components of a bogie, dampers are designed for six to eight years, which makes it rather difficult to acquire samples over the

entire lifetime via testing. A six to eight year, full system test of a railway vehicle is a very expensive operation and therefore a huge motivation to find alternatives for acquiring a valid training data set. Figure 3.3 provides an overview of different possibilities.

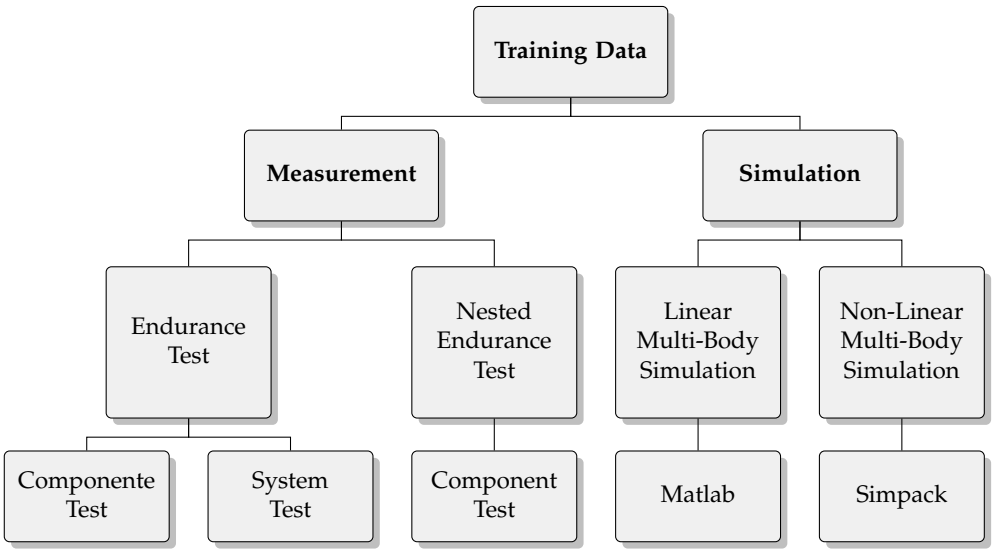


Figure 3.3.: Acquiring training data

Indeed a measured data set is the preferred source when it comes to model training. Either in form of an endurance test, which is not going to be possible for the most components because of their extended lifetime. Or a nested endurance test, which requires advanced lab equipment and will only be support single component tests. A complete system test will always deliver more reliable data than a nested test, however six to eight years of testing is not really an option. As more and more vehicles get equipped with a monitoring system, it may be possible in a few years, to use training data recorded during operation. As for now, simulated data sets are the common approach and will also be the chosen method within this thesis. Besides the cost and time factor, the flexibility in terms of system parameter variation is another advantage. Components properties can be varied at will, which guarantees a training data set with samples of every possible HS from new to broken condition. On the downside, simulated training data sets require knowledge about the

components deterioration behavior. Usually, there is more than just one failure mode, that leads to different evolution the the components system property. Most likely, it will be a combination of failure modes, which is responsible for the actual deterioration characteristics. This combination and the impact of the failure modes is in general unknown and can only be estimated with support of expert knowledge and previous component testing. For the damper deterioration the following failure modes have been considered:

- External sealing damage (Loss of oil)
- Internal sealing damage
- Valve damage
- Piston damage
- Rubber bearing embrittlement

All of them have a different impact on the damper characteristics when occurring. According to a statement of a damper supplier, external sealing damage and rubber bearing embrittlement are the most frequent failures. This damper degradation is going to be implemented in a three-dimensional, linear multi-body simulation, that is realized with Matlab. In contrast to a Simpack simulation, which has more sophisticated approaches for wheel-rail contact and friction phenomena and therefore a more realistic simulation, the solution with Matlab has way less computation time. Since the machine learning models are trained to react on a relative change of system properties, the absolute value of that property does not really matter. The process for training and testing will be the same, regardless of the inputs quality. The accuracy of the output however, can only be as accurate as the training data. This tradeoff between quality and computation time is accepted, because it will not effect the learning algorithm. Once completed, better input data can be processed in the exact same way.
For an artificially created training data set, several assumptions about the lifetime and the health evolution of the damper have to be made. In figure 3.2, those assumptions are represented by an input distribution $Dist(\boldsymbol{x})$. At this point it should only be said, that lifetime as well as health evolution are estimated with statistical distributions based on experience and previous test results. Together with a detailed description of the simulation model, those distributions are discussed in section 4.1.

3.3.2. Feature Selection

The railway vehicle consists of a car body and two bogies with two wheelsets each. It can be abstracted as a system of masses, springs and dampers, which are responsible for the dynamic system response. Therefore, the degradation of a component has a certain influence on the vehicle dynamics. Usually a modern railway vehicle is equipped with a large number of sensors, mostly acceleration sensors mounted on different areas on the vehicle. The feature selection is supposed to find the most informative and accurate quantity, derived from the observed sensor signals. The raw signal is converted to a time domain, frequency domain or time-frequency feature, that represents a components degradation. For the mechanical damper, these features are time based standard derivations within defined frequency bands [8].

A promising approach is presented in Girstmair *et.al.*[8], where a combination of heuristic and automated feature selection is applied. The heuristic method is a rough pre-selection, based on changes in the transfer function, which requires advanced domain knowledge. Due to the high number of possible sensor signals, a pre-selection is absolutely necessary to keep the computation time of the subsequent automated feature selection to a minimum. Automated selection methods are basically optimizing the output of a machine learning model by alternating the input quantities. This process of feature selection was applied to a railway vehicle in order to find an optimized set of features for the classification of secondary vertical dampers. The multi-body simulation can basically provide accelerations in x-, y- and z-direction for any given point on the vehicle. They serve as inputs for the feature selection. The result shows, that a combination of three informative features is enough to ensure an acceptable accuracy of the machine learning models. For the derivation of a RUL algorithm for the right sided, secondary suspended, vertical damper of the leading bogie, a total of three primary and secondary suspended, vertical acceleration standard deviations were selected. Those three quantities will serve as input observations for the machine learning algorithms. Together with the label “healthy” or “faulty”, they form a complete input data set $\mathcal{D} = \boldsymbol{x_1}, y_1, ..., \boldsymbol{x_N}, y_N$.

3.4. The Course of Action

The machine learning diagram (figure 3.2) is a good overview of what has to happen to successfully implement a prognostics algorithm. It can be pinned down that there are two phases, training and testing. Whereas training is the process to learn the model, testing is its actual application. As discussed in section 3.3, the training data is going to be simulated with a linear, three-dimensional multi-body model. Thereafter, the features mentioned in 3.3.2 are picked from the artificial training data set and are used to learn a SVM. Once learned, it is able to map the multi-dimensional feature input to a corresponding HS, hence a one-dimensional quantity. This output of the SVM will then serve as observation for the HMM, which is ultimately used to calculate the RUL. Figure 3.4 visualizes the course of action in three stages.

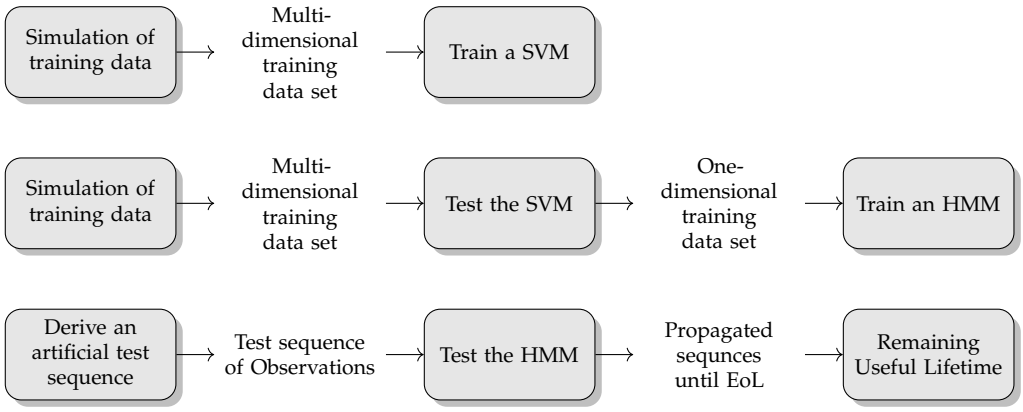


Figure 3.4.: The course of action to implement a RUL algorithm. All stages propagating horizontally. Stage 1 at the top, stage 2 in the middle and stage 3 at the bottom

Stage One A SVM is trained, using simulated healthy ($\widehat{=}$ 100% damping rate) and faulty ($\widehat{=}$ 0% damping rate) data sets.

Stage Two The SVM is tested with the intermediate damper stages, i.e. from 100% to 0% damping rate in 10% steps. Resulting in a mapping of the

dampers health condition to the three-dimensional feature input set. Thereafter, this results are concatenated in order to represent the entire lifetime of the damper. Additionally, a sequences of the same length with the corresponding health condition, by means of the damping rate determined by the simulation, is produced. Details are presented in section 4.2.3. The last step has statistical distributions underlying, which enables the generation of independent and different sequences of observations with corresponding HSs. Those concatenated, one-dimensional sequences serve as training data for the HMM.

Stage Three This stage describes the testing of the HMM. In course of the theoretical investigation of its performance, the actual testing process as it is performed with real data, has to be artificially recreated. In the real application the model is applied with observed data, predicting the RUL in daily or weekly intervals. Therefore, the provided data increases with every interval, until the component is broken. Naturally, the estimated RUL gets better in accuracy as the time goes on. This continuously increasing length of the applied data set is realized through an iterative approach, which provides an increasing percentage of the input sequence with every iteration. In reality, that sequence is observed by the health monitoring system. For the theoretical assessment within this thesis, this test sequence is synthetically created. Just like the training data set is produced in stage two, yet another sequence is derived, only without the label of the HS. An important property of the output sequences of stage two is, that all of them need to be independent from each other in order to get a unbiased result of the statistical models. For every partial sequence, the HMM should be in position to determine the current HS and predict its future evolution until the EoL. The provided length of the partial sequence, increases after every iteration. As more of the synthetic sequence is provided, the prediction is supposed to get better. Actual results are presented in chapter 5.

In conclusion, the course of action involves a multi-body simulation as well as training and testing of SVMs and HMMs. All assumptions and mathematics applied are presented in chapter 4.

4. Implemented Methods

The algorithm is designed to estimate the RUL of mechanical components of railway vehicles. The approach to establish that algorithm is independent from the predicted component, however the selected features and assumptions for training data are individually different. Despite the algorithm is implemented for the secondary vertical damper of the leading bogie on the right side, it can be used for any other component, provided that proper training data can be acquired.

When it comes to condition based and predictive maintenance, HMMs have been and still are in the focus of numerous research activities [16, 7, 17, 20, 19]. Many publications deal with similar tasks, namely the determination of the HS and the prediction of the RUL, which have already been specified as the main objectives of prognostic concepts in chapter 3. There exist several smart approaches, where the basically discrete and one-dimensional HMM is extended, to represent continuous observations and deal with coexisting deterioration modes. Since all of them are successful in predicting a mechanical components future condition, HMMs are chosen to be the approach for this application in railway industry [7, 16, 17].

According to the course of action described in section 3.4, a multi-body simulation as well as a SVM is necessary to process data, observed by a health monitoring system in a way, that an HMM can deal with it. Therefore, three methods are applied to develop a holistic algorithm, capable of predicting the RUL of a damper. This chapter provides theoretical foundations and details for those three methods. Their specific parameter settings are going to be subject of chapter 5.

4.1. A Linear Multi-Body Simulation

The objective is to synthetically generate training and testing data for SVMs and HMMs. Although nowadays many professional software packages, specialized on this kind of task are available, a simplified, linearized model is developed to effectively generate data. Being aware of the solutions imperfection, its advantage is a very short computation time, which is a rather good tradeoff. Considering that the generated data should simulate sensor observations, which are noisy anyway, the seemingly bad accuracy is no problem, because it gets lost in the noise. Within this section, a linear state-space model for a three-dimensional railway vehicle is derived, whose solution will be provided later on in chapter 5.

The desired output of a multi-body simulation is the system response to a certain excitation, represented by the equation of motion (EoM) for the center of gravity (CoG) of every participating component. For this special application, which involves the synthetic generation of healthy and faulty data sets, multiple simulations with varying system parameters are performed. A continuous variation of those parameter was initially considered, but discarded because of the enormously increased computation time. Additionally, the accuracy of the solution would have only improved slightly, compared with a discrete approach, assuming a smart choice of the iteration frequency. Along side the vehicle parameters, including the iterative changed damping characteristics, track parameters represent the second input for the simulation. Those are vertical (z) and lateral (y) excitations, as well as curvature and line routing. Those track conditions are taken into account by exciting the wheels according to the track data. The wheel-rail contact is due to its non-linearity a challenging topic. For this application the wheel profile is linearized which leads to a manageable system of equations. Further assumptions for the linear model are as follows [9]

- Small deflections and angles
- Small profile angle γ
- Rigidly mounted and flawless rail
- Constant velocity v_x and angular speed $\dot{\phi}$
- Linear traction law
- Neglected rotary slip

- Vertical wheel set acceleration $z \approx 0$ and angular wheel set acceleration $\ddot{\chi} \approx 0$

The simulation is done in the euclidean three-dimensional space $\mathbb{R}^3$. Although the model calculates in three dimensions, including curvature, line routing, lateral accelerations and all traction related phenomena which result in the sinusodial run (see section 4.1.3), a track of $40km$ length, is processed within a few minutes. Since the parameter variation, which simulates the deterioration of the damper is done iterative, a short computation time is absolutely necessary. The railway vehicle is abstracted according to figure 4.1, which leads to a model with 7 components and a total of 102 vehicle parameters, including geometrical dimensions, masses, moments of inertia, spring and damper characteristics as well as material specific parameters. Overall, the model has 23 EoM with just as much degrees of freedom (DoF). As it is the nature of common multi-body systems, masses are rigid bodies and springs and dampers do not have mass properties.

4.1.1. Data Acquisition Using a Multi-Body Simulation

A solution of a linearized multi-body model in Matlab is chosen over a Simpack simulation or something comparable. Reason for that is not least, because the generated data is used for analysis on subsequently performing statistical models, hence does not have the requirement to be most precisely. Therefore, the way faster computing, solution is preferred, which enables a quick and reproducible way of generating data, of sufficient accuracy. The iterative parameter variation includes only the considered secondary vertical damper, which is initiated with 100% of its nominal damping rat, and successively decreased by 10%, until its contribution is completely vanished, resulting in 11 iterations. Thereafter, each of those 11 damper settings, are computed with 11 randomly different gamma distributed revenue loads, resulting in a total number of 121 simulations. The course of action for the simulation is concluded as follows

1. Selection of the constant longitudinal velocity v_x
2. Definition of track related input data $\boldsymbol{u}$
3. Vehicle parameter settings
4. Definition of random revenue load

5. Derivation of system matrices for a linear state-space system
6. Solution of this linear system
7. Transformation of the EoM for the CoG to any given point on the vehicle
8. Saving the results as a specifically designed object
9. Decreasing the damper characteristic by 10%
10. Repeating the procedure from point 3) to 10) until damping rate has reached 0%

Whereas most of the points are straight forward and easy to implement, point 5) is actually a pretty hard task. Sections 4.1.2 and 4.1.3, deal with the definition of the system matrices. Assuming all of the above points are successfully completed, 121 data sets with the varying damping parameter and random revenue loads are produced. Each of them covering a distance of about $40km$, this results in a total of approximately $5 \cdot 10^3 km$, within which the dampers health condition goes linearly from completely healthy to completely faulty. First of all, this is a rather short distance, considering that the dampers lifetime will be around six to eight years. However, a direct simulation of several million kilometers is certainly not feasible. Second of all, a linear deterioration is very unlikely. Those two open points are very important assumptions, which have to be made in order to produce most realistic signal sequences. Therefore, two remaining tasks in order to acquire an appropriate data set are the following.

1. Define the maximum mileage that represents the dampers lifetime
2. Define the evolution of the dampers condition throughout its lifetime

Both of them will be estimated by a certain probability distribution, which will be described in section 4.2.3. The purpose of the multi-body simulation as such, will be met after the 121 simulation results are saved.

4.1.2. Model of a Railway Vehicle

A common railway vehicle is a rather complicated system. Figure 2.1 presents the bogie, which is from an engineering point of view, the most interesting component. Most of the vehicles functionality, such as track guiding, stabilization, braking or driving, is brought by sub-components of the bogie. To ensure stable and safe operation, it is common that the bogie is divided into

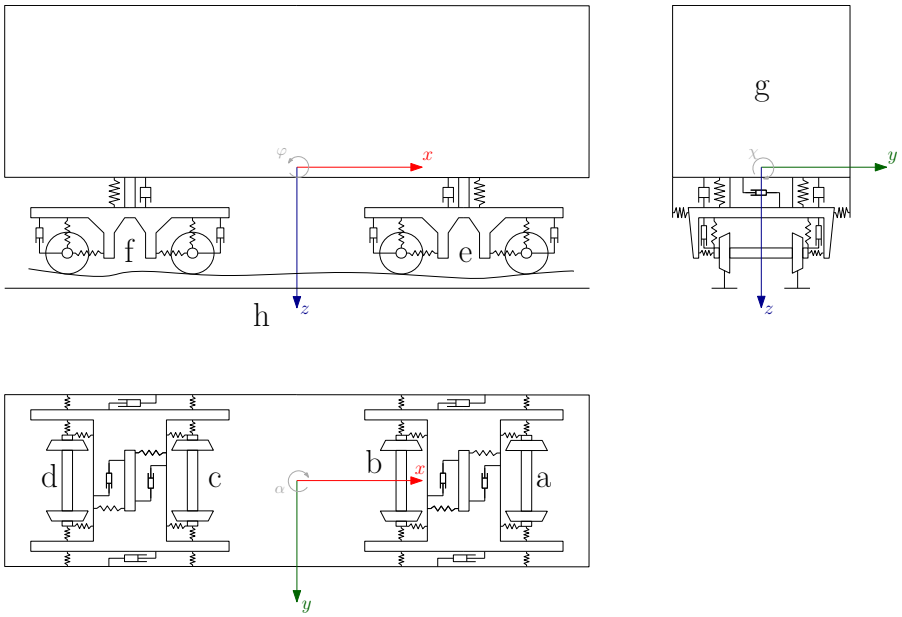


Figure 4.1.: A railway vehicle represented by a multi-body system: a, b, c, d) Wheel sets; e) Leading bogie; f) Trailing bogie; g) Car body, h) Track.

a primary and a secondary suspended level. The wheel sets however, can be considered as non-suspended, which leads to a total of three suspension levels within the vehicle. From the simulation models point of view, every single component contributes to the over all weight, but the effective mass is concentrated in the center of gravity of the corresponding suspension level. With respect to that, the vehicle is divided into the car body in the second level, two bogies with related parts in the first level and four wheel sets in the non-suspended level. Those seven components are connected with springs and dampers in longitudinal (x), lateral (y) and vertical (z) direction. The track defines the excitation for the multi-body system. Figure 4.1 shows the model which is used for the simulation. The plotted springs and dampers are not the only components contributing to the systems stiffness and damping rate. In fact, every component has certain stiffness and damping properties, which are considered as well.

Degrees of Freedom

A body in $\mathbb{R}^3$ without constraints has in general six DoF. Namely translation $\boldsymbol{x} = [x \quad y \quad z]^T$ as well as the corresponding rotations $\boldsymbol{\omega} = [\chi^{(x)} \quad \varphi^{(y)} \quad \alpha^{(z)}]^T$. Since one of the model assumptions is a constant velocity in driving direction, any EoM in x is unnecessary. The DoF decrease to five per body. For the wheel sets, the angular speed $\dot{\varphi}$ has a kinematic connection to the longitudinal motion. Therefore the rotation about the y-axis does not count as DoF either. Phenomena like the loss of contact between wheel and rail are not considered, which implies, under the assumption of the wheel as rigid body, that they directly follow the track excitation in z-direction. Therefore, a wheel sets motion in z direction is no DoF anymore. Both wheels are firmly mounted on the axle, which leads to another kinematic relation. The rotation about the x-axis is related to the left and right z excitation. That last constraint leaves the wheel set with only two DoF. In conclusion, four wheel sets à two DoF, plus two bogies and a car body à five DoF sum up to a total of the above stated 23 [9].

The EoM is solved with the "Newton-Euler Procedure", which will be reasoned in section 4.1.3. However, in course of this method, equations are established for every participating body and all their DoF. Hence, 23 EoM need to be identified in order to solve the problem. For a uniform representation of the equations, the following indices are used to address certain locations on the vehicle. Not explicitly mentioned is the index x, y and z which

represents the coordinate direction.

$$\text{Bogies :} \quad k(x) = \begin{cases} I & \text{if } x \geq 0 \\ II & \text{if } x < 0 \end{cases}$$

$$\text{Sides :} \quad j(y) \in \mathbb{N} = \begin{cases} 1 & \text{if } y \geq 0 \\ 2 & \text{if } y < 0 \end{cases}$$

$$\text{Axes :} \quad i(x,k) \in \mathbb{N} = \begin{cases} 0 & \text{if } x = 0,\ k = I \vee II \\ 1 & \text{if } x > 0,\ k = I \\ 2 & \text{if } x < 0,\ k = I \\ 3 & \text{if } x > 0,\ k = II \\ 4 & \text{if } x < 0,\ k = II \end{cases}$$

$$\text{Level :} \quad l = \begin{cases} u & \text{if non-suspended} \\ p & \text{if primary-suspended} \\ s & \text{if secondary-suspended} \end{cases}$$

$$\text{Origin :} \quad o = \begin{cases} c & \text{if spring} \\ d & \text{if damper} \\ ext & \text{if external} \end{cases}$$

Where the coordinates x and y measure from the considered components body-fixe coordinate system. With this indexing system, all quantities can be properly assigned to their location of impact. Since there are different use cases, the indexing is applied to each type of quantity exemplary.

$$\begin{aligned} \text{Forces and torques :} \quad & \psi_{Origin,\ Coordinate,\ Bogie,\ Axle,\ Side} \rightarrow \psi_{o,x,k,i,j} \\ \text{State variables :} \quad & \phi_{Level\ of\ Suspension,\ Bogie,\ Axle} \rightarrow \phi_{l,k,i} \\ \text{System parameters :} \quad & \phi_{Coordinate,\ Level\ of\ Suspension,\ Bogie,\ Axle,\ Side} \rightarrow \phi_{x,l,k,i,j} \\ \text{Geometrical quantities :} \quad & \Gamma_{Axle,\ Side} \rightarrow \Gamma_{i,j} \\ \text{Position vectors :} \quad & \Pi_{Coordinate,\ First\ Location,\ Second\ Location} \rightarrow \Pi_{x,kij,kij} \end{aligned}$$

Not all of them are always necessary to exactly address the right location, therefore k, i, or j will sometimes be left empty. Position vectors, which

are used to define relative locations on the components, are indexed with both positions. The vector points from the first, to the second index location. Basically vectorial quantities do not contain a coordinate index.
Considering the twenty-three DoF of the entire railway vehicle, the complete position vector $\boldsymbol{x}$ and the input vector $\boldsymbol{u}$, that represents the track related systems excitation, are defined as follows

$$\begin{aligned}\boldsymbol{x} = \big[&y_s \quad z_s \quad \varphi_s \quad \chi_s \quad \alpha_s \\ &y_{pI} \quad z_{pI} \quad \varphi_{pI} \quad \chi_{pI} \quad \alpha_{pI} \\ &y_{pII} \quad z_{pII} \quad \varphi_{pII} \quad \chi_{pII} \quad \alpha_{pII} \\ &y_{u1} \quad y_{u2} \quad y_{u3} \quad y_{u4} \\ &\alpha_{u1} \quad \alpha_{u2} \quad \alpha_{u3} \quad \alpha_{u4}\big]^T \quad ,\end{aligned} \tag{4.1}$$

$$\begin{aligned}\boldsymbol{u} = \big[&u_{y1} \quad u_{y2} \quad u_{y3} \quad u_{y4} \\ &u_{z1} \quad u_{z2} \quad u_{z3} \quad u_{z4} \\ &u_{\chi1} \quad u_{\chi2} \quad u_{\chi3} \quad u_{\chi4} \quad u_{\kappa}\big]^T \quad ,\end{aligned} \tag{4.2}$$

4.1.3. The Newton-Euler Procedure

This procedure was chosen because in comparison with energy-based principles like Lagrange, d'Alembert or Jourdain, the internal forces are part of the solution. Although it is not impossible with one of the other principles, it is associated with effort, which can be avoided with Newton-Euler. By considering every sub-component on its own, all the internal forces appear in the system equations. Since the intended parameter variation of the secondary vertical damper requires internal forces to be parameterized, this method is predestined. A rather simple example of the procedure is provided in appendix A.1.

It works with the two basic principles of Newton and Euler, which describe the dynamic of a body or a system of bodies with respect to their centers of gravity. The principle of linear and angular momentum is applied on every sup-component and relating to the body-fixed coordinate system, which has

its origin in the center of gravity.

$$\frac{d}{dt}\boldsymbol{p} = \frac{d}{dt}m\dot{\boldsymbol{x}} = m\ddot{\boldsymbol{x}} = \sum_{n=1}^{N} \boldsymbol{F}_n \quad , \tag{4.3}$$

$$\frac{d}{dt}\boldsymbol{L}^{(CoG)} = \frac{d}{dt}\boldsymbol{\theta}^{(CoG)}\dot{\boldsymbol{\omega}} = \boldsymbol{\theta}^{(CoG)}\ddot{\boldsymbol{\omega}} = \sum_{n=1}^{N} \boldsymbol{M}_n^{(CoG)} \quad . \tag{4.4}$$

Both equations are only valid for constant mass m. $\boldsymbol{p}$ is the linear momentum, $\boldsymbol{x}$ the linear position vector with its time derivatives $\dot{\boldsymbol{x}}$ and $\ddot{\boldsymbol{x}}$ with time t. $\boldsymbol{L}^{(CoG)}$ is the angular momentum with respect to the center of gravity, $\boldsymbol{\theta}^{(CoG)}$ the inertia tensor and $\boldsymbol{\omega}$ the angular position vector with its time derivatives $\dot{\boldsymbol{\omega}}$ and $\ddot{\boldsymbol{\omega}}$.

Equation 4.3 says that the temporal change of the linear momentum is equal to the sum of all effective forces $\boldsymbol{F}_n$ where N is the maximum number of forces on the considered body. For constant m this is Newton's second law. Equation 4.4 is also only valid for a body-fixed coordinate system and related to the center of gravity. It says that the temporal change of the angular momentum equals the sum of all effective torques $\boldsymbol{M}_n^{(CoG)}$[2, Chap. B, Sec. 3]. The above described principles are the basic approach to get to the EoM when using the Newton-Euler procedure. The desired outputs are as many equations as DoF in matrix form:

$$\boldsymbol{M}\ddot{\boldsymbol{x}} + \boldsymbol{D}\dot{\boldsymbol{x}} + \boldsymbol{C}\boldsymbol{x} = \boldsymbol{U}_d\dot{\boldsymbol{u}}(t) + \boldsymbol{U}_c\boldsymbol{u}(t) \quad . \tag{4.5}$$

This is a linear and time invariant differential equation of second order[1]. The system properties are represented through the masses and inertia $N \times N$ matrix $\boldsymbol{M}$, the $N \times N$ velocity proportional damping matrix $\boldsymbol{D}$ and the $N \times N$ position proportional stiffness matrix $\boldsymbol{C}$. The position vector and its first and second time derivative are of size $N \times 1$. On the right hand side is the system excitation, which is again sorted by their proportionality. The velocity proportional $N \times Q$ excitation matrix $\boldsymbol{U}_d$ and the $N \times Q$ position proportional share $\boldsymbol{U}_c$ define the impact of the input vector $\boldsymbol{u}$, which has the size $Q \times 1$. For a system of multiple bodies, N equals the number of bodies $k = 1, ..., K$, times their DoF, or in other words the total number of DoF of the entire system. Q

[1] Linear because the position vector and the excitation is linear. Time invariant because the system matrices are constant [2, Chap. O, Sec. 2]

depends on how many inputs are designated to represent the track excitation [2, Chap. O, Sec. 2]. In this simulation of a railway vehicle $N = 23$, which equals the number of DoF and $Q = 13$ because of the size of the excitation vector defined in 4.2. The best approach to fill all those system matrices, is to investigate every component separately, because the occurring forces are quite different. Only the internal connection forces as well as the centrifugal force appear to be part of every components equation. So before every components contribution to the system is determined separately, connection forces and torques as well as the generally effective centrifugal force, are defined.

The centrifugal force effects every sub-component and occurs when the vehicle drives through a curve. Usually the outer rail in the curve is raised by a certain amount which is called the "cant" $\tilde{u}$, in order to lower the lateral acceleration. If so, the weight force reduces the real lateral acceleration by its lateral component, which is determined by the amount of $\tilde{u}$. The actual centrifugal force is then

$$F_q = ma_q = m\Big(\frac{v^2}{R} - g\frac{\tilde{u}}{2b}\Big) = m\Big(u_\kappa v^2 - g\sin(\chi)\Big) \quad , \tag{4.6}$$

where m is the considered components mass and a_q its effective lateral acceleration, which is a function of the vehicle speed v and the track related parameters curve radius R, cant $\tilde{u}$ and the gauge exactness $2b$. Since u_κ is a better manageable track parameter and χ_{lki} will be included in the solution, the centrifugal is ultimately represented by those two quantities.

Connection forces are either related to a spring or a damper and manifest as a force or a torque, which are formulated as functions of the position vectors $\boldsymbol{x}$ and $\boldsymbol{\omega}$. As the coil spring force is proportional to its compression and the hydraulic damper force to its relative velocity, those can be written as:

$$\boldsymbol{F}_c^{(K,\hat{K})} = \int_{\boldsymbol{x}_K}^{\boldsymbol{x}_{\hat{K}}} \boldsymbol{c}(\boldsymbol{x})d\boldsymbol{x} \quad , \tag{4.7}$$

$$\boldsymbol{F}_d^{(K,\hat{K})} = \int_{\dot{\boldsymbol{x}}_K}^{\dot{\boldsymbol{x}}_{\hat{K}}} \boldsymbol{d}(\dot{\boldsymbol{x}})d\dot{\boldsymbol{x}} \quad . \tag{4.8}$$

The stiffness matrix $\boldsymbol{c}(\boldsymbol{x})$ defines the spring rate for each direction, which is

basically a function of its compression, i.e.,

$$\boldsymbol{c}(\boldsymbol{x}) = \begin{bmatrix} c_x(x) & 0 & 0 \\ 0 & c_y(y) & 0 \\ 0 & 0 & c_z(z) \end{bmatrix} \quad . \tag{4.9}$$

The damping matrix $\boldsymbol{d}(\dot{\boldsymbol{x}})$ defines the damping rate for each direction, which is also not constant but a function of its relative velocity, i.e.,

$$\boldsymbol{d}(\dot{\boldsymbol{x}}) = \begin{bmatrix} d_x(\dot{x}) & 0 & 0 \\ 0 & d_y(\dot{y}) & 0 \\ 0 & 0 & d_z(\dot{z}) \end{bmatrix} \quad . \tag{4.10}$$

$d\boldsymbol{x}$ is the differential of the linear position vector with its time derivative $d\dot{\boldsymbol{x}}$. The integral goes from the considered body K to the interacting body $\hat{K}$, both of which are connected through a spring or a damper.

The mixed terms are zero because the connection elements are assumed to transmit only in their main direction. Yet, a non-linear behavior can be expected from this kind of characteristics. Since the purpose of this simulation is to deliver quick estimates, this non-linearities are simplified to a linear function.

Spring and damper relative motion does not come only from translational motion, but also from rotation. Therefore the angular position vector contributes as follows

$$\Delta\boldsymbol{x} = \boldsymbol{\omega} \times \boldsymbol{l} \quad , \tag{4.11}$$

$$\Delta\dot{\boldsymbol{x}} = \frac{d}{dt}(\boldsymbol{\omega} \times \boldsymbol{l}) = \frac{d}{dt}\boldsymbol{\omega} \times \boldsymbol{l} + \boldsymbol{\omega} \times \frac{d}{dt}\boldsymbol{l} = \dot{\boldsymbol{\omega}} \times \boldsymbol{l} \quad , \tag{4.12}$$

where $\boldsymbol{l}$ is the distance vector between the CoG of the considered body and the respective spring or damper. Since the bodies are assumed to be rigid, $\boldsymbol{l} \neq f(t)$ and therefore drops out of equation 4.12. Superposition of the angular contribution and the translation leads to the total relative position and velocity

$$\tilde{\boldsymbol{x}} = \boldsymbol{x} + \Delta\boldsymbol{x} \quad , \tag{4.13}$$

$$\tilde{\dot{\boldsymbol{x}}} = \dot{\boldsymbol{x}} + \Delta\dot{\boldsymbol{x}} \quad . \tag{4.14}$$

With equations 4.7 to 4.14 applied on 4.3 and 4.4 and the linearity assumption, the resulting forces and torques (momentums) can be written as:

$$\begin{aligned} \boldsymbol{F}_c^{(K,\hat{K})} &= \boldsymbol{c} \int_{\tilde{\boldsymbol{x}}_K}^{\tilde{\boldsymbol{x}}_{\hat{K}}} d\tilde{\boldsymbol{x}} = \boldsymbol{c}(\tilde{\boldsymbol{x}}_{\hat{K}} - \tilde{\boldsymbol{x}}_K) = \\ &= \boldsymbol{c}\Big(\boldsymbol{x}_{\hat{K}} - \boldsymbol{x}_K + (\boldsymbol{\omega}_{\hat{K}} \times \boldsymbol{l}_{\hat{K}}) - (\boldsymbol{\omega}_K \times \boldsymbol{l}_K)\Big) \quad , \end{aligned} \tag{4.15}$$

$$\begin{aligned} \boldsymbol{M}_c^{(K,\hat{K})} &= \boldsymbol{l}_K \times \Big(\boldsymbol{c} \int_{\tilde{\boldsymbol{x}}_K}^{\tilde{\boldsymbol{x}}_{\hat{K}}} d\tilde{\boldsymbol{x}}\Big) = \boldsymbol{l}_K \times \Big(\boldsymbol{c}(\tilde{\boldsymbol{x}}_{\hat{K}} - \tilde{\boldsymbol{x}}_K)\Big) = \\ &= \boldsymbol{l}_K \times \Big(\boldsymbol{c}(\boldsymbol{x}_{\hat{K}} - \boldsymbol{x}_K + (\boldsymbol{\omega}_{\hat{K}} \times \boldsymbol{l}_{\hat{K}}) - (\boldsymbol{\omega}_K \times \boldsymbol{l}_K)\Big) \quad , \end{aligned} \tag{4.16}$$

$$\begin{aligned} \boldsymbol{F}_d^{(K,\hat{K})} &= \boldsymbol{d} \int_{\tilde{\dot{\boldsymbol{x}}}_K}^{\tilde{\dot{\boldsymbol{x}}}_{\hat{K}}} d\tilde{\dot{\boldsymbol{x}}} = \boldsymbol{d}(\tilde{\dot{\boldsymbol{x}}}_{\hat{K}} - \tilde{\dot{\boldsymbol{x}}}_K) = \\ &= \boldsymbol{d}\Big(\dot{\boldsymbol{x}}_{\hat{K}} - \dot{\boldsymbol{x}}_K + (\dot{\boldsymbol{\omega}}_{\hat{K}} \times \boldsymbol{l}_{\hat{K}}) - (\dot{\boldsymbol{\omega}}_K \times \boldsymbol{l}_K)\Big) \quad , \end{aligned} \tag{4.17}$$

$$\begin{aligned} \boldsymbol{M}_d^{(K,\hat{K})} &= \boldsymbol{l}_K \times \Big(\boldsymbol{d} \int_{\tilde{\dot{\boldsymbol{x}}}_K}^{\tilde{\dot{\boldsymbol{x}}}_{\hat{K}}} d\tilde{\dot{\boldsymbol{x}}}\Big) = \boldsymbol{l}_K \times \Big(\boldsymbol{d}(\tilde{\dot{\boldsymbol{x}}}_{\hat{K}} - \tilde{\dot{\boldsymbol{x}}}_K)\Big) = \\ &= \boldsymbol{l}_K \times \Big(\boldsymbol{d}(\dot{\boldsymbol{x}}_{\hat{K}} - \dot{\boldsymbol{x}}_K + (\dot{\boldsymbol{\omega}}_{\hat{K}} \times \boldsymbol{l}_{\hat{K}}) - (\dot{\boldsymbol{\omega}}_K \times \boldsymbol{l}_K)\Big) \quad . \end{aligned} \tag{4.18}$$

Whereby the relation between a force and a torque is defined by $\boldsymbol{M} = \boldsymbol{l} \times \boldsymbol{F}$. With these formulations of the connections, which are responsible for the interaction between bodies in a multi-body simulation, each individual component of the vehicle can be investigated separately. Note that the inner connection forces are effected by Newton's third law, *actio=reactio*, or $F_{ij}^K = -F_{ij}^{\hat{K}}$ [2, Chap. B, Sec. 3].

Wheel set

The dynamic of the wheel set is very important for the characteristic of the whole railway vehicle. The interaction between wheel and rail depends mainly on both geometries, speed and the prevailing friction conditions. Assuming a point-shaped contact, the oscillation of the vehicle is determined by the position of that contact point between wheel and rail. Figure 4.2 shows a non-linear wheelprofile and the occurring forces. T_{xij} and T_{yij} are slip forces, whose coordinate index relates to their origin, which is either longitudinal

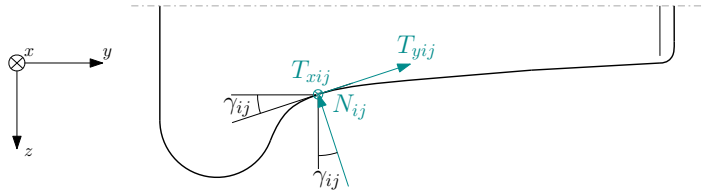


Figure 4.2.: Non-linear wheelprofile.

or lateral slip. The two of them, as well as the normal force depend on the profile angle $\gamma_{uij} = f(y_{ui})$, which is a function of the lateral movement. Both wheels and the axle have a rigid connection and therefore, they all have the same angular velocity $\dot{\varphi}_{ui}$. The motion of a wheel set depends on its relative position to the track, which can be described with the lateral displacement y_{ui} and the angular displacement α_{ui}. Therefor it is displaced by y_{ui} to the right, which increases lateral forces on the right and decreases them on the left side, because of the varying radius of the contact point. As a result, a lateral force pushes the wheel set back to the center of the track. In addition to that, the difference in contact radii on both sides and the fact that both wheels have the same $\dot{\varphi}_{ui}$, lead to longitudinal slip forces. They tend to brake down the wheel with the greater radius and accelerate the other one. This pair of forces results in a torque, which turns the wheel set by α_{ui}. Thereafter, the wheel set drives towards the center of the track, which is when the longitudinal slip forces vanish and the lateral forces are in balance. However, due to the angular displacement, lateral slip forces are induced, which drives the wheel set to the left side. It does not matter whether the wheel set gets displaced by y_{ui} or α_{ui} in the first place, this phenomena will always be the result. It is called sinusodial run and describes the motion of a a wheel set, which transmits the oscillation through the spring and damping connections to the bogie and thereafter to the vehicle body [9, 11]. Another contributor to the complex wheel rail interaction are rotary slips. They occur because the wheel, which is approximated by a cone, is forced to follow a straight line [11]. This rotary slip results in an additional slip torque, that is going to be neglected for simplicity reasons. Hence the relevant wheel set forces can be broken down to the following

- Connection forces (springs and dampers)
- Longitudinal slip forces

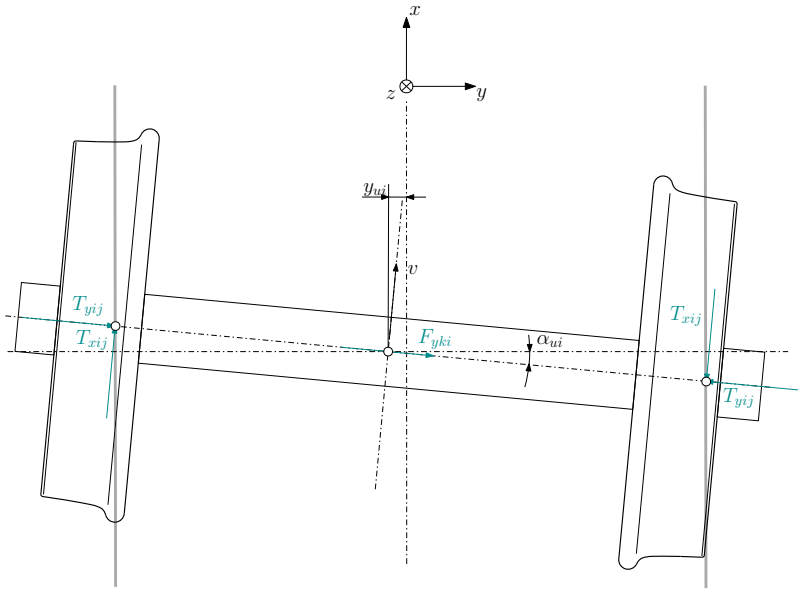


Figure 4.3.: Principle of track guiding.

- Lateral slip forces
- Normal forces between wheel and rail
- Centrifugal forces
- Weight forces

The only forces missing to build the EoM are the longitudinal and lateral slip forces. They are functions of the relative motion between wheel and rail (slip) as well as of the friction conditions. Kalker [13] found that longitudinal, lateral and rotational slip do not only effect the corresponding force, but also each other, which results in a highly non-linear function. For small slips however, those function can be linearized and lead to the following definition [9, 13]

$$T_{xij} = -f_{11} s_{xij} \quad , \tag{4.19}$$

$$T_{yij} = -f_{22} s_{yij} - f_{23} s_{\phi ij} \quad , \tag{4.20}$$

where f_{11} and f_{22} are adhesion coefficients and s_{xij}, s_{yij} and $s_{\phi ij}$ are slips. Since the simulation is based on the assumption that the rotary slip is neglected, $s_{\phi ij}$ drops out of equation 4.20. The adhesion coefficients are defined according to

Kalker [13] as

$$f_{11} = (a \cdot b) G C_{11} \quad , \tag{4.21}$$
$$f_{22} = (a \cdot b) G C_{22} \quad , \tag{4.22}$$

with a and b as the semi axis of the contact ellipse, G the modulus of shear and C_{11}, C_{22} the corresponding "Kalker" coefficients. For standardized wheel rail combinations, with respect to profile and material, those dimensions can be picked from certain tables [9].

Lateral and longitudinal slips are the relative motion in x and y between the wheel set and the rail. Any acceleration or deceleration is related to a slip, which in fact is required to make the motion even possible. For a railway vehicle, the longitudinal and lateral difference in velocity at the contact point can be defined as [9]

$$s_{xij} = \mp \frac{v - r_{ij}\dot{\varphi}_{ui} + b_i \dot{\alpha}_{ui}}{v} = \mp \frac{\tan(\gamma_{e,i})}{r_{0,ij}} (y_{ui} - u_{yi}) \mp \frac{b_i \dot{\alpha}_{ui}}{v} \quad , \tag{4.23}$$
$$s_{yij} = \frac{(\dot{y}_{ui} - \dot{u}_{ui}) - r_{ij}\dot{\varphi}_i \alpha_i}{v} \approx \frac{(\dot{y}_{ui} - \dot{u}_{ui})}{v} - \alpha_i \quad . \tag{4.24}$$

The $\mp$ represents the equation for the right side ($j = 1 \rightarrow$"-") and the left side ($j = 2 \rightarrow$"+") respectively. This notation will also be used in the following procedure. Equation 4.23 and 4.24 are based on the linearization of the wheel profile in a way, that the wavelength of the sinusodial run remains the same as it is for the non-linear profile. $\tan(\gamma_{e,i})$ is the equivalent conicity[2], $r_{0,i}$ the nominal wheel radius, r_{ij} the current radius and b_i the distance from the contact point to the wheel sets CoG.
With respect to the linearization, the contact wheel radius r_{ij}, the profile angle γ_{ij}, the equivalent conicity $tan(\gamma_{e,i})$ and the so called "contact angle parameter" ε_i are defined [9]. The latter basically helps later on to formulate

[2]The equivalent conicity defines the profile angle of an idealized conical wheel, so that the wavelength of the sinusodial run is the same, as of the real wheel profile [2]

well-arranged equations, i.e.,

$$r_{ij} = r_{0,ij} \pm \tan(\gamma_{e,ij})(y_{ui} - u_{ui}) \quad , \tag{4.25}$$

$$\gamma_{ij} = \gamma_{0,ij} \pm \varepsilon_i \frac{(y_{ui} - u_{ui})}{b_i} \quad , \tag{4.26}$$

$$\tan(\gamma_{e,i}) = \frac{r_{i,j=1} - r_{i,j=2}}{2(y_{ui} - u_{ui})} \quad , \tag{4.27}$$

$$\varepsilon_i = \frac{\gamma_{i,j=1} - \gamma_{i,j=2}}{2\frac{(y_{ui} - u_{ui})}{b_i}} \quad . \tag{4.28}$$

With those equations combined with 4.19 and 4.19, the slip forces can ultimately be formulated as

$$T_{xij} = -f_{11}\Big(\mp \frac{\tan(\gamma_{e,i})}{r_{0,i}}(y_{ui} - u_{ui}) \mp \frac{b_i \dot{\alpha}_i}{v}\Big) \quad , \tag{4.29}$$

$$T_{yij} = -\frac{f_{22}}{v}\left((\dot{y}_{ui} - \dot{u}_{ui}) - v\alpha_i\right) \quad . \tag{4.30}$$

Now the basic tools are introduced to bring the equations for linear and angular momentum into a form that can be processed further.

The wheel set has only two DoF, hence two EoM. Figure 4.4 shows the effective forces necessary to derive the equations. Geometrical dimensions are not illustrated in order to clearly represent the body. Equation 4.3 and 4.4 are applied on the wheel set. With respect to the indexing system defined above, equations are defined for the first wheel set of the leading bogie as follows:

$$\begin{aligned}
\sum F_y \quad &= m_{u1}\ddot{y}_{u1} \quad &&= T_{y11} + T_{y12} - N_{11}\sin(\gamma_{11}) + \\
&&&+ N_{12}\sin(\gamma_{12}) + F_{q,1} - F_{cyI1} - \\
&&&- F_{dyI1} + F_{y1,ext} \quad , \qquad (4.31)\\
\sum F_z \quad &= m_{u1}\ddot{z}_{u1} \approx 0 \quad &&= -N_{11} - N_{12} + F_{A,1} + F_{cz11} + \\
&&&+ F_{cz12} + F_{dz11} + F_{dz12} + F_{z1,ext} \quad , \qquad (4.32)\\
\sum M_x^{(CoG)} \quad &= \theta_{xxu1}\ddot{\chi}_{u1} \approx 0 \quad &&= b_1(N_{12} - N_{11}) + M_{cx11} - M_{cx12} + \\
&&&+ M_{dx11} - M_{dx12} + M_{x1,ext} \quad , \qquad (4.33)\\
\sum M_z^{(CoG)} \quad &= \theta_{zzu1}\ddot{\alpha}_{u1} \quad &&= b_1(T_{x12} - T_{x11}) + M_{cz11} - M_{cz12} + \\
&&&+ M_{dz11} - M_{dz12} + M_{z1,ext} \quad , \qquad (4.34)
\end{aligned}$$

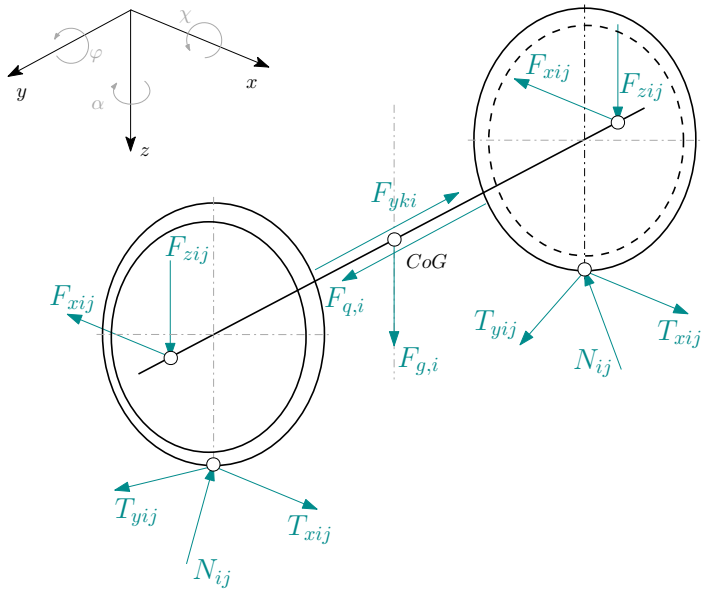


Figure 4.4.: Wheelset - effective forces.

The following assumptions are valid for the above set of equations. $T_y \ll N$, $\ddot{z}_{Wheelset} = \ddot{\chi}_{Wheelset} \approx 0$, $\chi \ll \gamma$, $\gamma \ll$ and therefore $\sin(\gamma) \approx \gamma$ and $\cos(\gamma) \approx 1$ [9]. With $F_{A,1}$ as the proportional net weight from the whole vehicle, which effects the considered wheel set. Note that the equilibrium condition in z and χ will not be influenced by the connection forces, because z and χ are not considered as DoF. Therefore, F_{cz}, F_{dz}, M_{cx} and M_{dx} are dropping out of equations 4.32 and 4.33. Within this simulation, the input excitation will not be realized through the external forces and torques. Rather than that, excitations will be induced as transverse wheel set movement in vertical u_{zi} and lateral u_{yi} direction as well as rotation about the x-Axis $u_{\chi i}$. Hence, all external forces and torques are dropping out of the above equation system. More to the excitation in chapter 4.1.4. Relating to the centrifugal force $F_{q,u1}$ the second term from equation 4.6 can be neglected because of the assumption of small angles. To dissolve the remaining connection forces and torques, the position vectors for both interacting components, from the respective CoG to the connections, are defined. In this case, interacting components are bogie I and wheel set 1. The connections are longitudinal and vertical at the axle

bearings as well as lateral in the middle of the axle. Therefore three position vectors for each interacting component can be defined as follows

$$\boldsymbol{l}_{I,I1} = \begin{bmatrix} l_{x,I,I1} \\ l_{y,I,I1} \\ l_{z,I,I1} \end{bmatrix} \qquad \boldsymbol{l}_{I,11} = \begin{bmatrix} l_{x,I,11} \\ l_{y,I,11} \\ l_{z,I,11} \end{bmatrix} \qquad \boldsymbol{l}_{I,12} = \begin{bmatrix} l_{x,I,12} \\ l_{y,I,12} \\ l_{z,I,12} \end{bmatrix} \tag{4.35}$$

$$\boldsymbol{l}_{1,I1} = \begin{bmatrix} l_{x,1,I1} \\ l_{y,1,I1} \\ l_{z,1,I1} \end{bmatrix} \qquad \boldsymbol{l}_{1,11} = \begin{bmatrix} l_{x,1,11} \\ l_{y,1,11} \\ l_{z,1,11} \end{bmatrix} \qquad \boldsymbol{l}_{1,12} = \begin{bmatrix} l_{x,1,12} \\ l_{y,1,12} \\ l_{z,1,12} \end{bmatrix} \tag{4.36}$$

From equations 4.32 and 4.33 with respect to all simplicity assumptions described above, it can be derived that $N_{11} = N_{12} = N = \frac{1}{2}F_{Ax,1}$. The definition of 4.28 and the normal forces lead to the so called "gravity spring stiffness" as in [9]

$$c_{RSyi} = \frac{F_{Ax,i}\varepsilon_i}{2b_i} \quad , \tag{4.37}$$

which enables a compressed representation of equation 4.38 and 4.39 in the form

$$\begin{aligned} m_{u1}\ddot{y}_1 &= T_{y11} + T_{y12} + N(\gamma_{12} - \gamma_{11}) + F_{q,1} - F_{cyI1} - F_{dyI1} = \\ &= -2\frac{f_{22}}{v}\left((\dot{y}_{ui} - \dot{u}_{ui}) - v\alpha_1\right) - F_{Ax,1}\,\varepsilon\,\frac{(y_{ui} - u_{ui})}{b_1} + \\ &\qquad + m_1 u_\kappa v^2 - F_{cyI1} - F_{dyI1} = \\ &= -2\frac{f_{22}}{v}\left((\dot{y}_{ui} - \dot{u}_{ui}) - v\alpha_1\right) - 2\,c_{RSy1}\,(y_{ui} - u_{ui}) + \\ &\qquad + m_1 u_\kappa v^2 - F_{cyI1} - F_{dyI1} \quad , \end{aligned} \tag{4.38}$$

$$\begin{aligned} \theta_{zzu1}\ddot{\alpha}_{u1} &= b_1(T_{x12} - T_{x11}) + M_{cz11} - M_{cz12} + M_{dz11} - M_{dz12} = \\ &= -2f_{11}b_1\Big(\frac{\gamma_{e,1}}{r_{0,1}}(y_{ui} - u_{ui}) + \frac{b_1\dot{\alpha}_1}{v}\Big) + \\ &\qquad + M_{cz11} - M_{cz12} + M_{dz11} - M_{dz12} \quad . \end{aligned} \tag{4.39}$$

To handle such equations within a numerical algorithm, they need to be in matrix form. Therefore, the lateral and rotary states are combined to the position vector $\boldsymbol{y}_{u1}$ as defined in equation 4.60, which means for the first wheel set of the leading bogie with its two DoF that $\boldsymbol{y}_{u1} := [y_{u1} \quad \alpha_{ui}]^T$. All occurring forces and torques need to be ordered by their contribution, by

means of velocity proportional, position proportional or external contributor. Equations 4.15 and 4.17 are used to dissolve the connections F_{cyI1}, F_{dyI1}, M_{cx11}, M_{cx12}, M_{dx11} and M_{dx12}. A detailed equation is provided in appendix A.2.

Considering the centered position of the lateral spring and damper connection between wheel set and bogie (see figure 4.4, F_{yIij}), the position vector $l_{1,I1} \equiv 0$, because the connection is assumed to be in the CoG. What remains as the representation of the lateral connection is

$$\begin{aligned}
F_{cyI1} &= c_{yI1}\Big(y_{pI} - y_{u1} + (\alpha_{pI} l_{x,I,I1} - \chi_{pI} l_{z,I,I1})\Big) \quad , \\
F_{dyI1} &= d_{yI1}\big(\dot{y}_{pI} - \dot{y}_{u1} + (\dot{\alpha}_{pI} l_{x,I,I1} - \dot{\chi}_{pI} l_{z,I,I1})\big) \quad , \\
M_{cz11} &= -c_{x11} l_{y,1,11}\Big((\varphi_{pI} l_{z,I,11} - \alpha_{pI} l_{y,I,11}) + \alpha_{u1} l_{y,1,11})\Big) \quad , \\
M_{cz12} &= -c_{x12} l_{y,1,12}\Big((\varphi_{pI} l_{z,I,12} - \alpha_{pI} l_{y,I,12}) + \alpha_{u1} l_{y,1,12})\Big) \quad , \\
M_{dz11} &= -d_{x11} l_{y,1,11}\Big((\dot{\varphi}_{pI} l_{z,I,11} - \dot{\alpha}_{pI} l_{y,I,11}) + \dot{\alpha}_{u1} l_{y,1,11}\Big) \quad , \\
M_{dz12} &= -d_{x12} l_{y,1,12}\Big((\dot{\varphi}_{pI} l_{z,I,12} - \dot{\alpha}_{pI} l_{y,I,12}) + \dot{\alpha}_{u1} l_{y,1,12}\Big) \quad .
\end{aligned}$$

Now the two EoM can be brought in matrix form, which results in the following equation system:

$$\begin{aligned}
&\begin{bmatrix} m_{u1} & 0 \\ 0 & \theta_{u1} \end{bmatrix} \cdot \begin{bmatrix} \ddot{y}_{u1} \\ \ddot{\alpha}_{u1} \end{bmatrix} + \begin{bmatrix} 2\frac{f_{22}}{v} - d_{yI1} & 0 \\ 0 & 2\frac{f_{11} b_1^2}{v} + d_{x11} l_{y,1,11}^2 - d_{x12} l_{y,1,12}^2 \end{bmatrix} \cdot \begin{bmatrix} \dot{y}_{u1} \\ \dot{\alpha}_{u1} \end{bmatrix} + \\
&\qquad + \begin{bmatrix} 2c_{RSy} - c_{yI1} & -2f_{22} \\ 2f_{11} b_1 \left(\frac{\gamma_{e,1}}{r_{0,1}}\right) & c_{x11} l_{y,1,11}^2 - c_{x12} l_{y,1,12}^2 \end{bmatrix} \cdot \begin{bmatrix} y_{u1} \\ \alpha_{u1} \end{bmatrix} = \\
&= \begin{bmatrix} 2\frac{f_{22}}{v} & 0 & 0 & m_{u1} v^2 \\ 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \dot{u}_{y1} \\ \dot{u}_{z1} \\ \dot{u}_{\chi 1} \\ \dot{u}_{\kappa} \end{bmatrix} + \begin{bmatrix} 2c_{RSy1} & 0 & 0 & 0 \\ 2f_{11} b_1 \left(\frac{\gamma_{e,1}}{r_{0,1}}\right) & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} u_{y_1} \\ u_{z_1} \\ u_{\chi 1} \\ u_{\kappa} \end{bmatrix}
\end{aligned} \tag{4.40}$$

This is the desired form of the EoM for the wheel set, where system parameters are represented by the mass matrix $\boldsymbol{M}_{ui}$, the damping matrix $\boldsymbol{D}_{ui}$ and the stiffness matrix $\boldsymbol{C}_{ui}$ [2, Chap. O, Sec. 2]. The excitation is on the right hand side of the equation, represented by the velocity proportional input matrix $\boldsymbol{U}_d$

and the position proportional matrix $\boldsymbol{U}_c$. A general notation for this equations is

$$\boldsymbol{M}_{ui}\ddot{\boldsymbol{x}}_{ui} + \boldsymbol{D}_{ui}\dot{\boldsymbol{x}}_{ui} + \boldsymbol{C}_{ui}\boldsymbol{x}_{ui} = \boldsymbol{U}_{di}\dot{\boldsymbol{u}}_i(t) + \boldsymbol{U}_{ci}\boldsymbol{u}_i(t) \quad . \tag{4.41}$$

All terms from the connection forces and torques, that contain state variables from any other than the considered body, are included in the system equations of the corresponding component. That is how the interaction between components impacts the EoM of the entire system. To solve the equation, the Newton Euler procedure has to be done for all sub-components. A comparison of equation 4.41 and 4.5 shows that the wheel set's equation has the same form as the desired equation for the entire system. That means, once every sub-components contributing system equation is determined, the entire system can easily be formulated.

Bogie

The bogie is not as complex as the wheel set. There are no traction forces or non-linearities that need to be considered in order to get a good estimate of the EoM. All forces considered in this simulation, are related to a connection to either one of the other sub-components, or an excitation. The equilibrium condition contains all interacting connection forces, which can be properly formulated as a function of position or velocity with equation 4.15 to 4.18. The result will very much look like equation 4.5 from the wheel set, however the dimension of the matrix will correlate to the DoF. The bogie has five, because the x-direction is determined by the constant input velocity of the vehicle and therefore not part of the simulation. Figure 4.5 shows the relevant connections, which will again be represented by forces and torques from springs and dampers. The equilibrium conditions 4.3 and 4.4 are exemplary applied to the leading bogie I. The sub-components position vector is defined as:

$$\boldsymbol{x}_{pI} = \begin{bmatrix} y_{pI} & z_{pI} & \varphi_{pI} & \chi_{pI} & \alpha_{pI} \end{bmatrix} \quad . \tag{4.42}$$

$$\tag{4.43}$$

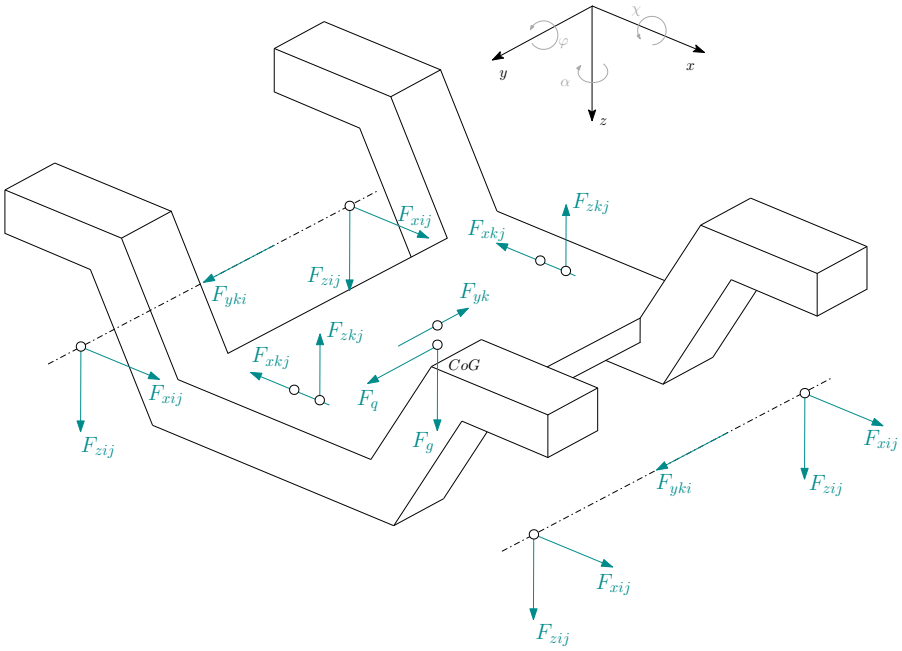


Figure 4.5.: Bogie - effective forces.

The resulting five equations for the leading bogie are:

$$\begin{aligned}\sum F_y \quad &= m_{pI}\ddot{y}_{pI} \quad &&= -F_{cyI} - F_{dyI} + F_{cyI1} + F_{dyI1} + \\ &&&+ F_{cyI2} + F_{dyI2} + F_{q,pI} \quad , \qquad (4.44)\\ \sum F_z \quad &= m_{pI}\ddot{z}_{pI} \quad &&= -F_{czI01} - F_{dzI01} - F_{czI02} - F_{dzI02} + \\ &&&+ F_{cz11} + F_{dz11} + F_{cz12} + F_{dz12} + \\ &&&+ F_{cz21} + F_{dz21} + F_{cz22} + F_{dz22} + F_{g,I} \quad , \qquad (4.45)\end{aligned}$$

$$\begin{aligned}
\sum M_x^{(CoG)} = \theta_{xxpI}\ddot{\chi}_{pI} \quad &= -M_{cxI01} - M_{dxI01} + M_{cxI02} + M_{dxI01} + \\
&+ M_{cx11} + M_{dx11} - M_{cx12} - M_{dx12} + \\
&+ M_{cx21} + M_{dx21} - M_{cx22} - M_{dx22} - \\
&- M_{cxI} - M_{dxI} + M_{cxI1} + M_{dxI1} + \\
&+ M_{cxI2} + M_{dxI2} \quad , \qquad (4.46)\\
\sum M_y^{(CoG)} = \theta_{yypI}\ddot{\varphi}_{pI} \quad &= -M_{cyI01} - M_{dyI01} - M_{cyI02} - M_{dyI02} - \\
&- M_{dyI001} - M_{dyI001} + \\
&+ M_{cyI11} + M_{dyI11} + M_{cyI12} + M_{dyI12} + \\
&+ M_{cyI21} + M_{dyI21} + M_{cyI22} + M_{dyI22} \quad , \qquad (4.47)\\
\sum M_z^{(CoG)} = \theta_{zzpI}\ddot{\alpha}_{pI} \quad &= M_{czI1} + M_{dzI1} - M_{czI2} - M_{dzI2} + \\
&+ M_{czI01} + M_{dzI01} - M_{czI02} - M_{dzI02} - \\
&- M_{cz11} - M_{dz11} + M_{cz12} + M_{dz12} - \\
&- M_{cz21} - M_{dz21} + M_{cz22} + M_{dz22} \quad . \qquad (4.48)
\end{aligned}$$

In equation 4.47, two torques, namely M_{dyI001}, and M_{dyI002}, can not be properly assigned to a location with the provided indexing system. The reason is, because at those locations, spring and damper do not effect the same point on the body. The secondary vertical damper is slightly moved, which is why both those "special" torques only occur in relation with a damper. The secondary vertical spring is in line with the CoG, hence no torque contribution about the y-axis.

By applying equations 4.15 to 4.18 to the equilibrium conditions, the connection forces can be dissolved in the same way as it was demonstrated for the wheel set. The result is a rather large set of of equations, that can be represented in the quite familiar matrix form as follows

$$\boldsymbol{M}_{pI}\ddot{\boldsymbol{x}}_{pI} + \boldsymbol{D}_{pI}\dot{\boldsymbol{x}}_{pI} + \boldsymbol{C}_{pI}\boldsymbol{x}_{pI} = \boldsymbol{U}_{dpI}\dot{\boldsymbol{u}}_{pI}(t) + \boldsymbol{U}_{cpI}\boldsymbol{u}_{pI}(t) \quad . \qquad (4.49)$$

The dimension of the system matrices $\boldsymbol{M}$, $\boldsymbol{D}$ and $\boldsymbol{C}$ is 5×5, since the bogie has five DoF. In the presented form, the equations can be combined with all the other components at the end.

Car body

Since the car body is very similar to the bogie in terms of their representation within the multi-body simulation, the procedure to get the system equations is exactly the same. Only connection forces and torques as well as the centrifugal force are applied to the car body. It interacts with both bogies and is considered to be in the second level of suspension. However, as an abstracted contributor to the dynamic system, it can be treated equally. The position vector is defined as

$$\boldsymbol{x}_s = \begin{bmatrix} y_s & z_s & \varphi_s & \chi_s & \alpha_s \end{bmatrix} \quad . \tag{4.50}$$

(4.51)

Again equilibrium conditions are applied to formulate the five contributing

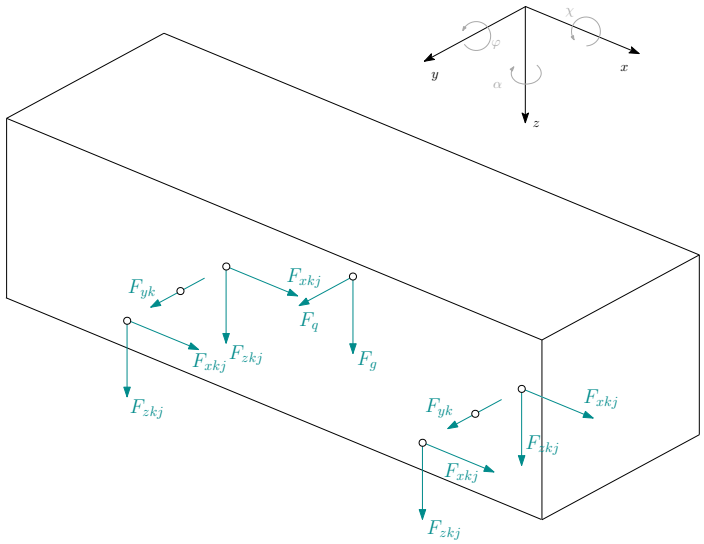


Figure 4.6.: Car body - effective forces.

system equations as:

$$\sum F_y \quad = m_s \ddot{y}_s \quad = -F_{cyI} - F_{dyI} + F_{cyII} + F_{dyII} + F_{q,s} \quad , \tag{4.52}$$

$$\sum F_z \quad = m_s \ddot{z}_s \quad = F_{czI01} + F_{dzI01} + F_{czI02} + F_{dzI02} + \\ + F_{czII01} + F_{dzII01} + F_{czII02} + F_{dzII02} + F_{g,s} \quad , \tag{4.53}$$

$$\begin{aligned}\sum M_x^{(CoG)} = \theta_{xxs}\ddot{\chi}_s \quad &= -M_{cxI} - M_{dxI} - M_{cxII} - M_{dxII} + \\ &+ M_{cxI01} + M_{dxI01} - M_{cxI02} - M_{dxI02} + \\ &+ M_{cxII01} + M_{dxII01} - M_{cxII02} - M_{dxII02} \quad , \qquad (4.54)\\ \sum M_y^{(CoG)} = \theta_{yys}\ddot{\varphi}_s \quad &= -M_{cyI01} - M_{dyI01} - M_{cyI02} - M_{dyI02} + \\ &+ M_{cyII01} + M_{dyII01} + M_{cyII02} + M_{dyII02} \quad , \qquad (4.55)\\ \sum M_z^{(CoG)} = \theta_{zzs}\ddot{\alpha}_s \quad &= M_{czI} + M_{dzI} - M_{czII} - M_{dzII} - \\ &- M_{czI01} - M_{dzI01} + M_{czI02} + M_{dzI02} - \\ &- M_{czII01} - M_{dzII01} + M_{czII02} + M_{dzII02} \quad . \qquad (4.56)\end{aligned}$$

And once more, the vectorial evaluation of the above stated equations with 4.15 to 4.18, delivers the desired output

$$\boldsymbol{M}_s\ddot{\boldsymbol{x}}_s + \boldsymbol{D}_s\dot{\boldsymbol{x}}_s + \boldsymbol{C}_s\boldsymbol{x}_s = \boldsymbol{U}_{ds}\dot{\boldsymbol{u}}_s(t) + \boldsymbol{U}_{cs}\boldsymbol{u}_s(t) \quad . \qquad (4.57)$$

The dimension of the system matrices is equal to equation 4.49 of the bogie.

Now that all sub-components are investigated, their system matrices can be combined in order to get a system of equations for the entire vehicle. Four wheel sets, two bogies and a car body are the contributors. Ultimately, twenty-three equations are provided, that have to be solved in order to get a solution.

4.1.4. Numerical Solution of the Equation System

The derived EoM are linear differential equations of second order. However, most of the existing numerical methods require a differential equation of first order

$$\dot{\boldsymbol{y}} = \boldsymbol{F}(\boldsymbol{y}, t) \quad . \qquad (4.58)$$

To convert the existing EoM to a system of first order, a state-space transformation is performed. The basic idea is to replace the vector triple

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix} , \qquad \dot{\boldsymbol{x}} = \begin{bmatrix} \dot{x}_1 \\ \vdots \\ \dot{x}_N \end{bmatrix} , \qquad \ddot{\boldsymbol{x}} = \begin{bmatrix} \ddot{x}_1 \\ \vdots \\ \ddot{x}_N \end{bmatrix} ,$$

with the state vector and its time derivative

$$\boldsymbol{y} = \begin{bmatrix} x_1 \\ \vdots \\ x_N \\ \dot{x}_1 \\ \vdots \\ \dot{x}_N \end{bmatrix} , \qquad \dot{\boldsymbol{y}} = \begin{bmatrix} \dot{x}_1 \\ \vdots \\ \dot{x}_N \\ \ddot{x}_1 \\ \vdots \\ \ddot{x}_N \end{bmatrix} .$$

Therefore, the system of 23 equations of second order gets transformed into a system of 46 equations of first order. A nice tradeoff, because several numerical methods exist, that can conveniently solve these kind of equations [14]. The only thing left to do, is to rearrange all system matrices and excitation vectors in order to preserve the equilibrium conditions, based on which the EoM was derived in the first place. The desired form of the entire system of equations is as follows:

$$\dot{\boldsymbol{y}} = \boldsymbol{A}\boldsymbol{y} + \boldsymbol{B}\boldsymbol{u}(t) \quad , \tag{4.59}$$

where $\boldsymbol{A}$ and $\boldsymbol{B}$ represent the new state matrices, that have to be found and $\boldsymbol{y}$ is the full state vector of the 23 DoF railway vehicle. With the definition of $\boldsymbol{x}$ in equation 4.1, that state vector is

$$\boldsymbol{y} = \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \quad \text{and} \quad \dot{\boldsymbol{y}} = \begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} \quad . \tag{4.60}$$

With that knowledge, the composition of the new state matrices can be derived. Multiplied by the state vector, they have to deliver the same result as the previously found EoM. In order to replace $\boldsymbol{x}$ with $\boldsymbol{y}$, the equation is rearranged as follows:

$$\begin{aligned} \boldsymbol{M}\ddot{\boldsymbol{x}} &= -\boldsymbol{D}\dot{\boldsymbol{x}} - \boldsymbol{C}\boldsymbol{x} + \boldsymbol{U}_d\dot{\boldsymbol{u}}(t) + \boldsymbol{U}_c\boldsymbol{u}(t) \\ \ddot{\boldsymbol{x}} &= -\boldsymbol{M}^{-1}\boldsymbol{D}\dot{\boldsymbol{x}} - \boldsymbol{M}^{-1}\boldsymbol{C}\boldsymbol{x} + \boldsymbol{M}^{-1}\boldsymbol{U}_d\dot{\boldsymbol{u}}(t) + \boldsymbol{M}^{-1}\boldsymbol{U}_c\boldsymbol{u}(t) \\ \ddot{\boldsymbol{x}} &= -\boldsymbol{D}_m\dot{\boldsymbol{x}} - \boldsymbol{C}_m\boldsymbol{x} + \boldsymbol{U}_{dm}\dot{\boldsymbol{u}}(t) + \boldsymbol{U}_{cm}\boldsymbol{u}(t) \quad . \end{aligned} \tag{4.61}$$

Now the position vector can be replaced by the state vector and equation 4.61 is rearranged to get a system in the form of 4.59. Moreover, the position and velocity excitation vectors have to be concluded to $\boldsymbol{u} = [u \quad \dot{u}]^T$. An additional equation is obtained, however the order is reduced which enables

the application of well known solvers. The system matrices are ultimately becoming

$$\boldsymbol{A} = \begin{bmatrix} 0 & \dots & 0 & 1 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & 0 & \dots & 1 \\ & \boldsymbol{C}_m & & & \boldsymbol{D}_m & \end{bmatrix}, \qquad \boldsymbol{B} = \begin{bmatrix} 0 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & 0 & \dots & 0 \\ & \boldsymbol{U}_{cm} & & & \boldsymbol{U}_{dm} & \end{bmatrix}.$$

Where the dimension of $\boldsymbol{A}$ is $\mathbb{R}^{(2N \times 2N)}$ and of $\boldsymbol{B}$ it is $\mathbb{R}^{(2Q \times 2Q)}$. For the presented railway vehicle $2N = 46$ and $2Q = 26$.

Now all necessary equations to define a linear state-space model are provided. Such a model can conveniently be solved by any commercial software package that supports matrix operations, which is shown in chapter 5. It is in the nature of the Newton-Euler procedure that the resulting equations describe the dynamics of the components CoG. In order to produce a proper training data set for the subsequently performed statistical models, all acceleration signals that usually would have been recorded by a health monitoring system, are evaluated as well. That is realized through translatory and rotatory transformation matrices, that make use of the known geometries. Since the components are perfectly rigid bodies, this transformation is purely geometrical, hence a rather easy task to solve. Like mentioned in section 3.3.2, the statistical models preferably use features, such as the standard derivation or the mean value, which is why those quantities are evaluated as well. In conclusion, the solution of the derived state-space model is the state vector, consisting of position, velocity and acceleration of all twenty-three DoF for any given point on the vehicle. Together with the additionally calculated statistical quantities and some meta data about the system parameters, these variables are saved as the result of one iteration. Hence, the data acquisition by the multi-body simulation is complete, and the next step in the course of action follows.

4.2. Classification with Support Vector Machines

The procedure presented in this thesis, uses the classifier to convert a multi-dimensional to a one-dimensional data set, which correlates to a certain health condition of the considered component. The one-dimensional quantity is required by the subsequently performed HMM. Therefore not only the class, but also a certain quantifying measure is of note. A SVM provides this measure in form of a posterior probability, that indicates how likely a data point is classified as it is. Thereafter, the posterior probability derived from the simulated input data, is concatenated to a sequence, that represents the dampers entire lifetime. Length and composition[3] of that sequence will be defined in section 4.2.3.

A SVM is a kernel based statistical model, with the purpose of classifying data. When an input data set is presented, it is often of interest, to which class, type or group a certain sample belongs. Inputs are usually a selection of statistically representative features, which can for example be selected according to a method described in section 3.3.2. With proper features and a well trained SVM, pattern recognition is possible, even with a small number of samples. This, and its highly appreciated generalization abilities are its advantages in comparison with conventional classification methods. SVMs work with kernels, which can be interpreted as a mathematical space of the dimension d. If the input data is not linearly separable in the current feature space, the so called "kernal trick" is applied. The data x is transformed into a higher dimensional space, which then allow a linear separation of data. In this particular space, the SVM algorithms are executed with the result of a "hyperplane" which separates the occurring classes of feature data. Although SVMs work as one-class up to multi-class recognition tools, for its broad application within the field of CBM two-classes are used [24].

The common objective in the area of CBM is to determine the HS of a certain mechanical component. For now, the classes are defined as healthy and faulty, which can be represented by a two-class SVM. Within the mathematical formulation of the model, the classes correspond to the label value $y =$

[3] In that context, the composition describes how long the component is assumed to stay in each HS. In other words, it defines the relative number of samples which are used for signals of the corresponding HS.

$+1$ or $y = -1$. The classification can be achieved by providing a specific data set, that contains features with the corresponding label of the HS. This data is provided by a multi-body simulation described in section 4.1. SVM algorithms for learning are basically available within the libraries of most programming software, like Matlab or Python. However, the learning process will be described roughly to point out the advantages of SVMs.

4.2.1. Learning of a Support Vector Machine

The following notation is used to pin down the mathematics of SVMs. X is considered the input space, that contains the input data set $\boldsymbol{x}_i \in \mathbb{R}^d$ with $i = 1, ..., N \in \mathbb{N}$, in a d-dimensional space. A Kernel is defined by $\mathcal{K}(\boldsymbol{x}, \boldsymbol{z}) = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{z}) \rangle$, if the scalar product $\forall \boldsymbol{x}, \boldsymbol{z} \in X$ and the mapping $\phi : X \to F$ both exist [21]. F is the feature space, that could be of any dimension, which is basically dependent on the type of kernel.

The above stated mathematics concerning kernels can be interpreted as methodology to transform or "map" the input space X to a higher-dimensional feature space F. This mapping can be applied to enable a linear separation of data, which might not have been possible in the input space. Basically there is no upper bound of the feature space dimension, because it does not effect the computation time of the algorithm. The conclusion is, that there will always be a feature space F, where data can be separated linearly by a hyperplane. Mapped back to the input space, that hyperplane is most likely not linear, which enables the separation of basically not linearly separable data. SVMs can process multi-dimensional input, however in the spirit of clarity, just two input features are used to describe the way, how that hyperplane is determined. Figure 4.7 shows the hyperplane that separates the positive samples o, from the negative samples x, under the assumption of separable data. It is quite intuitive that the best separating plane is the one with the biggest margin, since this is the width that a data sample can change without being missclassified. Therefore the maximum width is used as a decision for picking the final hypothesis of the plane. The hyperplane has the equation

$$\boldsymbol{w}^T \boldsymbol{x} + b = 0 \quad , \tag{4.62}$$

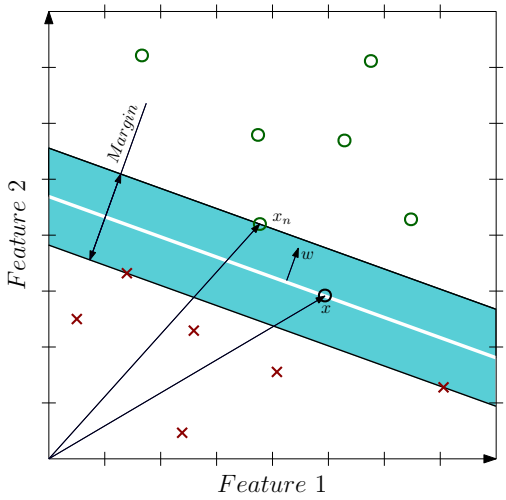


Figure 4.7.: Maximum margin hyperplane for a two-class SVM.

where $\boldsymbol{w}$ represents a vector, normal to the plane and b is the bias. For each o-sample of the data set, the corresponding class is $y_i = +1$, for each x-sample it is $y_i = -1$. Therefore the class of the data point can be determined as follows:

$$y_i = sign(\boldsymbol{w}^T \boldsymbol{x}_i + b) \quad . \tag{4.63}$$

For any data sample, this equation can also be written as [24]

$$\boldsymbol{w}^T \boldsymbol{x}_i + b = \begin{cases} > 0 & \text{if } y_i = +1 \\ < 0 & \text{if } y_i = -1 \end{cases} \quad . \tag{4.64}$$

Since it does not affect the hyperplane whether $\boldsymbol{w}^T$ is large or small, it can be normalized to accomplish scale invariance. Without any loss of generalization, $\boldsymbol{w}^T$ can be scaled in order to get the equation for positive and negative samples $\boldsymbol{x}_n$, which are right on the margin, i.e.:

$$|\boldsymbol{w}^T \boldsymbol{x}_n + b| = 1 \quad . \tag{4.65}$$

The norm is applied to make the equation valid for both classes. To calculate the actual margin, the vector $\boldsymbol{x}_n$ on the margin is subtracted by a generic

point on the hyperplane $\boldsymbol{x}_n$ and thereafter projected to the norm of the normal vector $\|\boldsymbol{w}\|$. With respect to equation 4.62 and 4.65, the margin is

$$\begin{aligned} Margin &= \frac{\boldsymbol{w}^T}{\|\boldsymbol{w}\|}|\boldsymbol{x}_n - \boldsymbol{x}| = \\ &= \frac{1}{\|\boldsymbol{w}\|}|\boldsymbol{w}^T\boldsymbol{x}_n - \boldsymbol{w}^T\boldsymbol{x}| = \\ &= \frac{1}{\|\boldsymbol{w}\|}|\boldsymbol{w}^T\boldsymbol{x}_n + b - \boldsymbol{w}^T\boldsymbol{x} - b| = \\ &= \frac{1}{\|\boldsymbol{w}\|} \quad . \end{aligned} \tag{4.66}$$

This is actually half the margin, but since the positive and negative side are symmetric and equal in size, it leads to the same result. So this is the quantity that has to be maximized in order to provide a decision for the best hypothesis. That maximization has to happen with respect to equation 4.65, because the margin is supposed to be maximized within the closest positive and negative sample:

$$\begin{aligned} &\text{Maximize:} \quad \frac{1}{\|\boldsymbol{w}\|} \\ &\text{Subject to:} \quad \min_{n=1,\dots,N} |\boldsymbol{w}^T\boldsymbol{x}_n + b| = 1 \quad . \end{aligned}$$

Both equations are mathematically inconvenient to solve. Therefore the maximization is replaced by a minimization and the minimum of the absolute value over all samples is taken care of as follows:

$$\begin{aligned} &\text{Minimize:} \quad \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} \\ &\text{Subject to:} \quad y_n(\boldsymbol{w}^T\boldsymbol{x}_n + b) \geq 1 \qquad \forall n \quad . \end{aligned}$$

The absolute value is replaced by the assumption that every data point is classified correctly, which implies a compensation of the equations sign, because positive samples have $y_n = +1$ and negative samples $y_n = -1$. The required minimum is taken care of by the $\geq$ condition. This minimization problem with an inequality constraint still needs some minor changes in order to be solvable. Hence, the "Karush-Kuhn-Tucker" (KKT) conditions are used

to derive the Lagrangian as follows [24, 21]

$$\mathcal{L}(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} - \sum_{n=1}^{N} \alpha_n (y_n(\boldsymbol{w}^T\boldsymbol{x}_n + b) - 1) \quad , \tag{4.67}$$

where $\alpha_n \geq 0$, $\forall n$ are the positive Lagrange multipliers, which are introduced by KKT to change the inequality constraint to an equality constraint. To pursue the minimization problem, the derivatives of $\mathcal{L}$ with respect to $\boldsymbol{w}$ and b are evaluated and thereafter substituted back to equation 4.67.

$$\nabla_{\boldsymbol{w}}\mathcal{L} = \boldsymbol{w} - \sum_{n=1}^{N} \alpha_n y_n \boldsymbol{x}_n = \boldsymbol{0} \, , \qquad \frac{\partial \mathcal{L}}{\partial b} = -\sum_{n=1}^{N} \alpha_n y_n = 0 \, , \tag{4.68}$$

$$\mathcal{L}(\boldsymbol{\alpha}) = \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n,m=1}^{N} y_n y_m \alpha_n \alpha_m \boldsymbol{x}_n^T \boldsymbol{x}_m \quad . \tag{4.69}$$

Now this formulation can be solved by minimizing with respect to $\boldsymbol{\alpha}$ and with subject to the KKT condition $\alpha_n \geq 0$, $\forall n$ and the equation on the right side of 4.2.1. This can effectively be achieved by quadratic programming algorithms, which deliver the solution for $\boldsymbol{\alpha}$. Analyzing the KKT condition $\alpha_n(y_n(\boldsymbol{w}^T\boldsymbol{x}_n + b) - 1) = 0$ and reviewing the minimization criterion results in the conclusion that if $\alpha_n > 0$, the vector $\boldsymbol{x}_n$ has to be directly on the outer boundary of the margin. The reason is, because only then, the second term of the KKT condition becomes 0, which is required for it to hold. Thus, $\boldsymbol{x}_n$ is called a support vector (SV). In figure 4.7 for example, the hyperplane is build with three SVs. By rearranging the left equation of 4.2.1 and the KKT condition, the hyperplane parameters can be determined

$$\boldsymbol{w} = \sum_{SV} = \alpha_n y_n \boldsymbol{x}_n \quad , \tag{4.70}$$

$$b = \frac{1}{y_n} - \boldsymbol{w}^T \boldsymbol{x}_n \quad . \tag{4.71}$$

So for any data set of dimension d, $\boldsymbol{w}$ is computed through a summation over all SVs with $\boldsymbol{\alpha}$ as parameters. Therefore its dimension is reduced dramatically, from d to the number of SVs. This is good news for the generalization of the model, because the out-of-sample error decreases for a decreasing number of SVs and an increasing number of samples N [1]. Equation 4.71 can be

evaluated with any one of the SVs with its corresponding label. Hence, all necessary parameters are defined in order to use equation 4.63 on a new data set to perform classification.

Note that the entire procedure takes place in the feature space F. If the input space X is not linearly separable, the mapping $\phi : X \to F$ will transform data into a higher-dimensional space. Howsoever this space may look like, the input for quadratic programming will be very similar to equation 4.69, i.e.:

$$\mathcal{L}(\boldsymbol{\alpha}) = \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n,m=1}^{N} y_n y_m \alpha_n \alpha_m \phi^T(\boldsymbol{x}_n)\phi(\boldsymbol{x}_m) \quad . \tag{4.72}$$

Even if the mapping increases the data dimension significantly, it will always be an inner product that has to be solved. Therefore, the dimension of the problem solved by quadratic programming, depends only on the size of the input data set in terms of samples N. This is another advantage, because the increased dimension enables the solution of highly non-linear problems, without loosing any computation performance. A transformation back to the input space, would deliver a "hyperplane", that is not a plane anymore. In

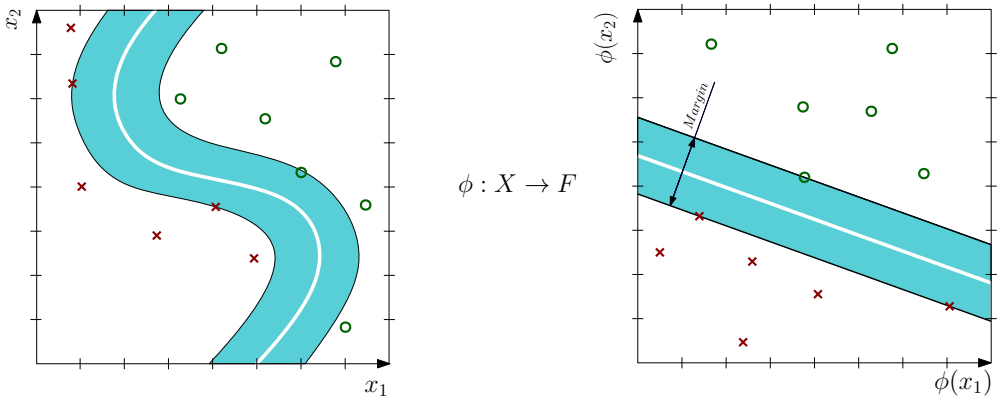


Figure 4.8.: Kernel transformation from input space X to feature space F.

conclusion, the mathematical consideration shows, that a SVM can handle non-linear data sets with good generalization ability. High-dimensional input does not necessarily mean overfitting and low-dimensional input has still a good chance of a low out-of-sample error. Due to the fact that the mathematics

behind a SVM use the inner vector product to compute the output, the kernel function can transform the input to a high-dimensional output and take advantage of that, without increasing computation time. All of those properties make SVMs a popular method with a broad field of application.

4.2.2. Testing a Support Vector Machine

The statistical model trained with a labeled data set, is able to classify completely new and unlabeled data. The output according to equation 4.63 is either $y_i = +1$ for "positive" samples and $y_i = -1$ for "negative" samples. In addition to that, it is possible to calculate the so called "score" $s \in \mathbb{R}$ of every sample within the data set. It represents the distance to the hyperplane in the F space and is therefore an indication for the probability of a sample, being correctly classified. The higher the score, the further away is the sample from the separating hyperplane. However, it is just an indication of the actual probability of correct classification. To calculate the latter, a posterior probability function $P(s)$ has to be fit to the SVM. This function depends on whether the data set is perfectly separable or not and on the nature of the SVM in terms of types of classes. Figure 4.9 shows possible $P(s)$ for a two-class SVM. The

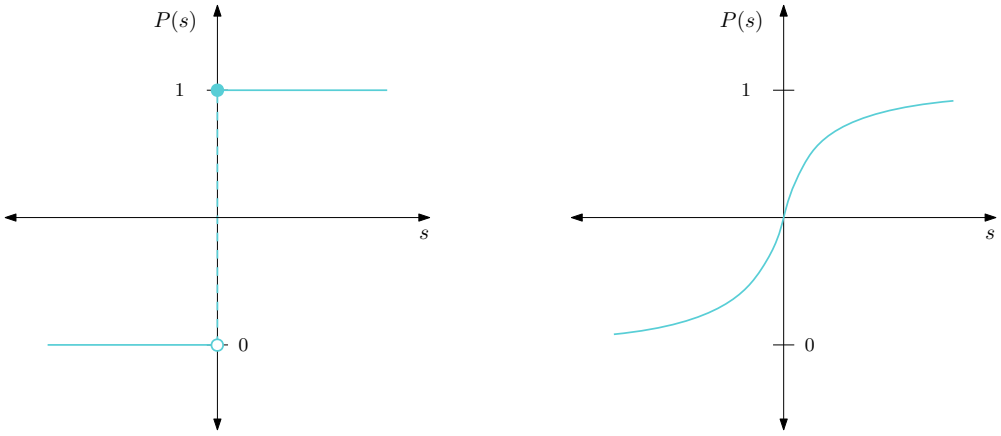


Figure 4.9.: Fit of a posterior probability to the score by: a) Step function; b) Sigmoid function.

step function is used to fit the posterior probability if the data set is perfectly separable. Otherwise, the Sigmoid function is applied. The output of the SVM is then between 0 and 1, which represents the probability of the considered sample to be in the positive class. The closer the considered sample gets to the hyperplane, the higher is the chance for misclassification. However, the posterior probability is a quantity that can be well interpreted. For most cases of health monitoring of mechanical components, the Sigmoid function will be used to fit it. Hence, a continuous transfer from the 100% positive sample to the 100% negative sample will be provided, which enables conclusion about the intermediate HS of the component. The lower the posterior probability, the better the health condition.

The posterior probability is basically the quantity, used to assign the features to a corresponding HS. Once the SVM is trained with 100% healthy ("positive" samples) and 0% healthy ("negative" samples), it is tested with the intermediate conditions to acquire a mapping of the HS. The next section provides details about how the obtained posterior probabilities are concatenated in order to get the desired sequence.

4.2.3. Length and Composition of the Synthetic Sequence

As already mentioned in section 4.1.1, there are two crucial points that need to be clarified in order to produce a sequence that represents the entire lifetime of the considered component.

1. Define the maximum mileage that represents the dampers lifetime
2. Define the evolution of the dampers condition throughout its lifetime

Usually this kind of assumptions are made based on experience or measurements. Unfortunately measuring the characteristics of mechanical components in railway vehicles is a huge effort and not part of any existing maintenance process. In some very rare cases, the damping characteristics where recorded during the dampers lifetime. However, the quality of those results is rather poor and the quantity is not quite statistically representable. Moreover, none of the tested dampers were broken or close to a foreseeable failure, which means that the degradation in damping rate can only be assessed for the

achieved mileage. Nevertheless, those measurements were used to derive a possible statement for point two of the above stated.

Evolution of the Damping Condition

A slight decrease of approximately 20% in damping rate could be observed for all investigated parts. That decrease happened within the first $0.2 \cdot 10^5 km$ of operation. After this, the damping rate remained basically stable until the dampers got substituted by new parts. Unfortunately a certain variance from the nominal damping rate of about $\pm 10\%$ also contributes to the already bad data situation. However, the initially, slightly degrading characteristics, followed by a plateau is expected to take a sudden end. After a certain mileage, the deterioration will decrease rapidly. Figure 4.10 shows how these findings can be applied to the assumption of the HS evolution. A Poisson

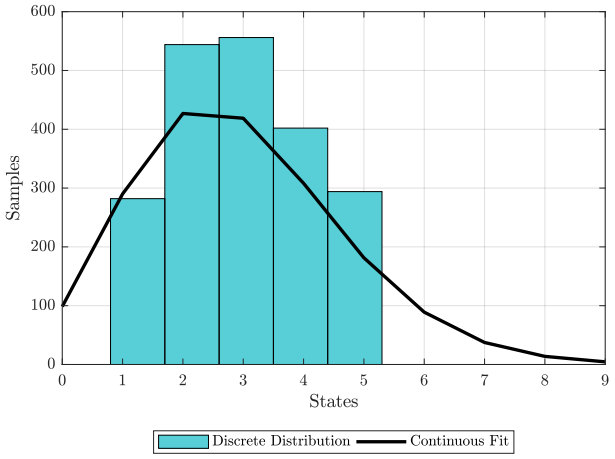


Figure 4.10.: Poisson distribution of health state evolution

distribution $\mathcal{P}(\lambda)$, with $\lambda = \frac{1}{3}N$, where N is the number of states, is chosen to represent the initial decrease and the subsequent stable scope, followed by the rather quick deterioration until the complete brake down. Figure 4.10 shows exemplary the distribution for a sequence if the number of states N is chosen

to be five. Within the simulation however, the damping rate was decreased by 10%, which results in 10 distinct HS. A proper mapping of the simulated HSs to the desired output HSs is required. The y-axis in figure 4.10 gives the actual number of samples simulated with each mapped HS. One sample represents the calculated mean value from one iteration of the simulation, which equals approximately one day of operation. This downsizing of the data set is absolutely necessary, because otherwise over-fitting[4] could happen easily. So the evolution is estimated by the Poisson distribution, but the absolute value of simulated mileage is still unclear.

Total Mileage of a Damper

For this problem, actually no existing measurements can be consulted. It would be necessary to observe accelerations during the complete lifetime of a component, which can easily reach six years and more. Obviously such testing is not reasonably, but unfortunately highly appreciated. Different approaches are presented in section 3.3.1, however for this synthetic data sequence a normal distribution will be assumed. Figure 4.11 shows the assumption that the dampers EoL is standard distributed $\mathcal{N}(\mu, \sigma)$. The estimated mean was chosen according to the absolute maximum mileage that a damper is currently expected to be in operation, which is $\mu = 1.6 \cdot 10^6\ km$. A rather high standard derivation of $\sigma = 2.0 \cdot 10^5 km$ is assumed, in order to represent diverse failure modes. Figure 4.11 exemplary shows ten samples which are drawn from the distribution. The x-axis shows the actual mileage values, that will serve as maximum mileage for the corresponding damper simulation, the y-axis relates to the absolute number of samples which are drawn.

In conclusion, the desired sequence is a labeled data set, which contains posterior probabilities, as well as the corresponding HSs. In other words, for every evaluated point, there should be a sequence of posterior probabilities and a sequence of HSs. The length of this sequences will be represented by the normal distribution $\mathcal{N}(\mu, \sigma)$, its composition by the Poisson distribution $\mathcal{P}(\lambda)$. Based on those distributions the posterior probabilities are concatenated to a sequence that represents the components complete life cycle. Although it

[4] A phenomena within statistics, where to much data is used for learning. The trained model fits the training data perfectly, but does not generalize at all.

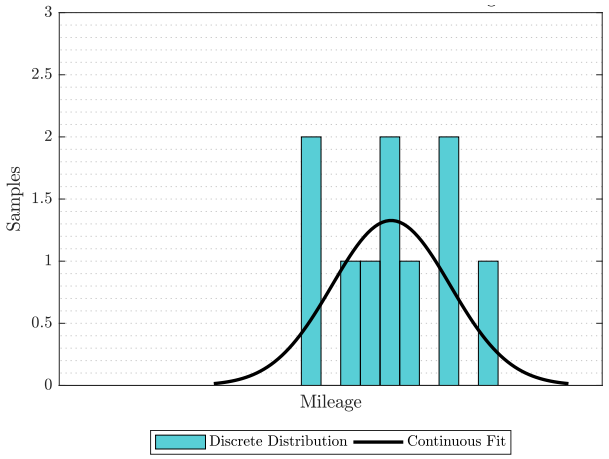


Figure 4.11.: Normal distributed maximum mileage of a damper

seems like a simple accumulation of the results, it is not, because firstly, the revenue load defined during the simulation causes some noise and secondly, the posterior probabilities of the SVMs are also related to a Sigmoid distribution. Therefore, different posterior probabilities can be produced for the same HS, depending on the Sigmoid parameters, which are basically defined during the training phase of the SVM. Hence, the actual output is viable, providing a sequence for the posterior probabilities and one for the HS. This method is used to generate training data as well as testing data, for the implemented statistical models.

4.3. Prognostics with Hidden Markov Models

This section provides the required tools in terms of HMMs, to develop a proper RUL algorithm. A previously simulated data set, that represents the entire lifetime of a damper, is generated with a multi-body simulation and a subsequent SVM. In that manner, several independent data sets representing the source for training and testing are produced. Ultimately, the HMM with all its implemented features is trained and tested with different sequences from

those generated data sets. To evaluate its out-of-sample error. The following section presents the results and the detailed parametrization of the used tools.

HMMs are commonly known for their excellent performance in speech recognition and deterioration estimation. Basically they model real-life processes, with a number of distinct finite states $\boldsymbol{S} = \{S_1, S_2, ..., S_N\}$ with $N \in \mathbb{N}$ and distinct finite observations $\boldsymbol{V} = \{V_1, V_2, ..., V_K\}$ with $K \in \mathbb{N}$. The latter is the observed or measured quantity, which is emitted by the currently active and unknown or "hidden" state, hence the models name. Generally speaking, they are statistical models that describe a two staged stochastic processes, which is repeatedly executed, forming a corresponding state sequence $\boldsymbol{q} = (q_1, q_2, ..., q_T)$ and observation sequence $\boldsymbol{o} = (o_1, o_2, ..., o_T)$ with $T \in \mathbb{N}$ [6, 19]. The path through the model from time step $t = 1$ to $t = T$, is a chain of discrete and causal events, where the frequency of the time discretization depends on the considered problem. Each time step within the sequence, consists of a two staged stochastic process. The first stage deals with the states, the second one with the emitted observations. HMMs are statistical models, that capture transitions between states and their emitted observations between time steps, with probabilities or probabilistic distributions. An HMM is a probabilistic, finite automaton of states and observations, where every state has both, an emission probability for every possible observation $b_j(k) = P(o_t = V_k | q_t = S_j)$, as well as a transition probability for each pair of states $a_{ij} = P(q_t = S_j | q_{t-1} = S_i)$ [19]. At this point, two fundamental assumption have already been made, namely the "markov property" and the "output independence assumption".

- Markov property [6]:
  The current state probability q_t depends only on the immediate predecessor, not on the entire sequence, i.e.

$$P(q_t | q_1, ..., q_{t-1}) = P(q_t | q_{t-1}) \quad . \tag{4.73}$$

- Output independence assumption [6]:
  Each time step is associated with one state, which emits one observation. The probability distribution for that observation depends only on the state from which it was emitted, i.e.

$$P(o_t | o_1, ..., o_{t-1}, q_1, ..., q_t) = P(o_t | q_t) \quad . \tag{4.74}$$

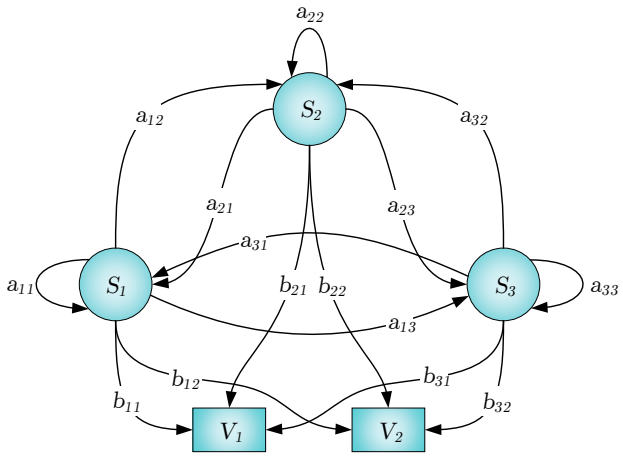


Figure 4.12.: HMM Automaton with three distinct states and two distinct observations.

Figure 4.12 is a generic example of such an automaton for three distinct states S_1, S_2 and S_3, as well as two distinct observation symbols V_1 and V_2. For an HMM described by such an automaton, a_{ij} are entries of the transition matrix $\boldsymbol{A}$ of size $N \times N$ with the property $\sum_{j=1}^{N} a_{ij} = 1, \forall i$. $b_j(k)$ are entries of the emission matrix $\boldsymbol{B}$ of size $N \times K$ with the property $\sum_{k=1}^{K} b_j(k) = 1, \forall j$. The definition of an HMM is completed with the initial state distribution $\boldsymbol{\pi} = P(q_1 = S_i)$ where $1 \leq i \leq N$. Thereby a compact notation of the parameters of an HMM is given by

$$\lambda = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi}) \quad , \tag{4.75}$$

which requires implicitly a definition of N and K [20]. Figure 4.13 shows how states and observations are linked with probabilities within the time series. For every HMM, not the hidden states, but the emitted observations are the measured quantities, which are very often continuous signals, like sound levels or accelerations. However, the basic version of an HMM uses a discrete and finite observation space, which allows the emission probabilities to be presented in matrix form, just like in the above example. That leads to the distinction between “discrete” and “continuous” HMMs. Although

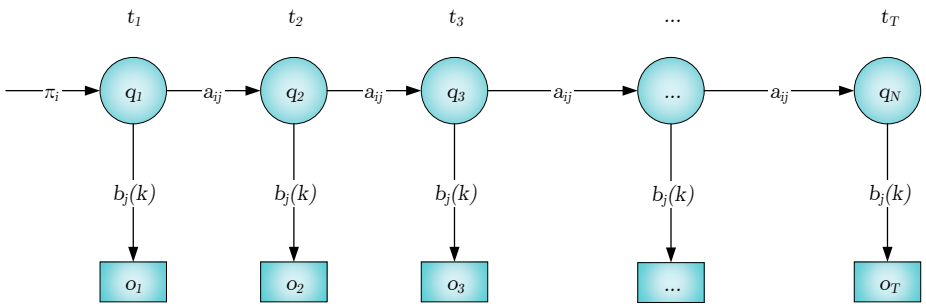


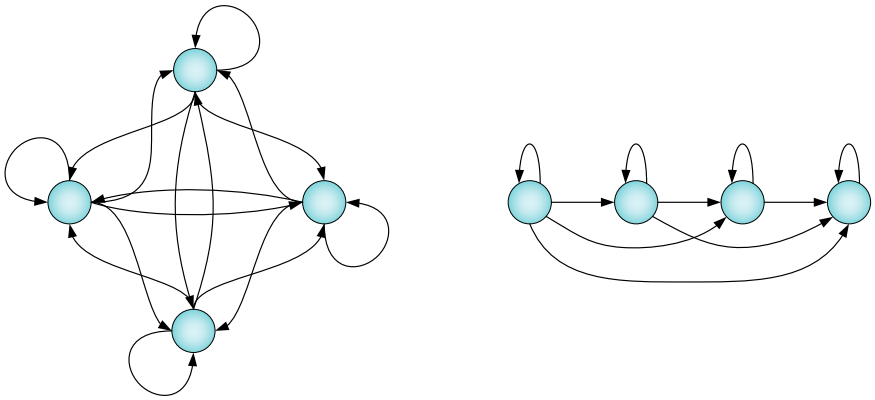
Figure 4.13.: HMM Sequence of states and observations as a time series.

the continuous HMM is the better choice for deterioration problems, the discrete model is implemented, because many of the applied algorithms exist in Matlab libraries, yet only for the discrete HMMs. Hence, the basic version of the developed RUL algorithm will refer to a discrete HMM. Another distinction can be made between types of automatons in terms of their state connections, i.e. which transitions are basically possible $(a_{ij} > 0)$ and which are not $(a_{ij} = 0)$. Figure 4.14 showes two different types of HMMs [20]. The model 4.14(a) is a so called "ergodic-HMM", which allows transitions between every distinct state. Model 4.14(b) is a left-right HMM, which basically only allows increasing states. Since this type can model a developing process, it is chosen to be implemented in course of the deterioration prediction algorithm. In addition to the left-right constraint, transitions are only allowed to the direct successor state, because the degradation of mechanical components is assumed to happen continuously. Therefore, the transition probabilities will be of the following form, considering the indices being positive integers, i.e.

$$a_{ij} \begin{cases} > 0 & \text{if } i \leq j \leq i+1 \\ = 0 & \text{otherwise} \quad . \end{cases} \tag{4.76}$$

The theoretical basics presented in the following section do not consider any particular type of HMM, hence they are valid for all types. The theory facilitates the understanding of how the algorithm is capable of satisfying the two major requirements of prognostics

1. Detection of the current HS



(a) (b)

Figure 4.14.: Possible state transitions for different types of HMMs. (a) 4 state ergodic HMM; (b) 4 state left-right HMM.

2. Prediction of its future evolution (RUL).

Since HMMs are probabilistic models, a definition of the "Conditional Probability", the "Law of Total Probability" as well as "Bayes' Theorem" are necessary, in order to understand the mathematics behind the algorithm.

$$P(A|B) = \frac{P(A,B)}{P(B)} \quad , \tag{4.77}$$

$$P(B) = \sum_{A \in U} P(B|A) \cdot P(A) \quad , \tag{4.78}$$

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \quad . \tag{4.79}$$

Where A and B are discrete random events within the state space U, $0 \leq P(A) \leq 1$ and $0 \leq P(B) \leq 1$ their prior probabilities and $P(A|B)$ the conditional probability of A given B [15].

4.3.1. The Three Problems of Hidden Markov Models

This section includes a lot of basic mathematics for HMMs. It mainly relates to the work of R. Rabiner [20], who gives a holistic tutorial on HMMs. Some mathematical formulations however, are inherited from A. Fink [6] and R. Movellan [19].

In the previous section, all parameters of an HMM were defined by $\lambda = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi})$, which are now ready to be applied. There are basically three use cases for HMMs, each of them using the defined statistical relations between states, observations and their corresponding sequences, in a different way in terms of input and output.

1. The evaluation problem:
   Give the model $\lambda = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi})$ and an observation sequence $\boldsymbol{o} = (o_1, ..., o_T)$. Objective is to calculate the likelihood of that sequence $P(\boldsymbol{o}|\lambda)$ [20].
2. The decoding problem:
   Give the model $\lambda = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi})$ and an observation sequence $\boldsymbol{o} = (o_1, ..., o_T)$. Objective is to calculate the sequence of states $\boldsymbol{q}^* = (q_1^*, ..., q_T^*)$, which best explains the observation sequence $\boldsymbol{o}$. [20].
3. The learning problem:
   Give an observations sequence $\boldsymbol{o} = (o_1, ..., o_T)$. Objective is to maximize the probability $P(\boldsymbol{o}|\lambda)$ by adjusting the model parameters $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{\pi}$ [20].

The second requirement of prognostics, which relates to the estimation of the RUL, can not be solved by the standard applications for HMMs alone. Still, they are necessary for the HS detection and in a way, they form the base for the RUL computation, which is discussed in section 4.3.3.

The Evaluation Problem

The evaluation algorithm determines how well a given sequence fits to a given model, or in other words, how likely the given sequence is produced by the given model. It can be used to find the best model among others, or to quantify a certain model [20]. Therefore, the objective is to determine $P(\boldsymbol{o}|\lambda)$ with respect to the markov property and the output independence assumption,

which are represented through equations 4.73 and 4.74. Considering the HMM's two staged stochastic process, including the transition and emission probabilities and the basic statistical relations from equation 4.77 and 4.78, the probability of a given sequence can be derived through direct calculation as follows [20]:

$$\begin{aligned} P(\boldsymbol{o}|\lambda) &= \sum_{K} P(\boldsymbol{o}, \boldsymbol{q}^K|\lambda) \\ &= \sum_{K} P(\boldsymbol{o}|\boldsymbol{q}^K, \lambda) \cdot P(\boldsymbol{q}^K|\lambda) \\ &= \sum_{\boldsymbol{q}^K} \pi_{q_1} b_{q_1}(o_1) a_{q_1,q_2} b_{q_2}(o_2) \cdot ... \cdot a_{q_{T-1},q_T} b_{q_T}(o_T) \quad . \end{aligned} \tag{4.80}$$

The above equation points out, that a summation over all possible state sequences $\boldsymbol{q}^K$ is necessary. Figure 4.15 helps to understand the propagation through the trellis, which represents all possible states q_i at each time step t, given the sequence of emissions $\boldsymbol{o}$. States are drawn as circles, observations as squares and the edges represent the transition probability $P(q_t|q_{t-1}) = a_{ij}$. Since this method would require operations on the order of $\approx 2TN^T$ [20], it is not feasible at all, which motivates the research for a better algorithm. The so called **Forward Algorithm** is a more effective method to compute the likelihood of the observation sequence, given the model. It relates to the markov property, which says, that the path of states taken to reach the current state i is irrelevant for the probability of the predecessor state j. Therefore, in the successor time step $t+1$, only all possible states of step t are considered. That enables a parallel computation of all possible states paths [6]. The auxiliary forward variable, which represents the possibility of being in state S_i at time t, given the previous sequence of observations until t, is defined as:

$$\alpha_t(i) = P(o_1, o_2, ..., o_t, q_t = S_i|\lambda) \quad . \tag{4.81}$$

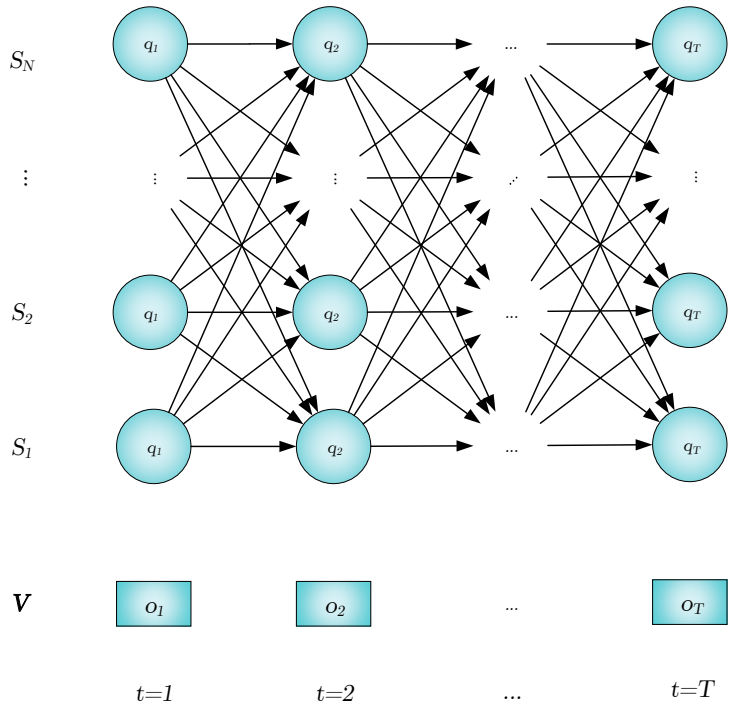


Figure 4.15.: State space trellis of states for a given observation sequence.

With this definition, the desired probability $P(\boldsymbol{o}|\lambda)$ can be computed by inductively solving for $\alpha_t(i)$ [20].

$$\text{Initialization}: \quad \alpha_1(i) = \pi_i b_i(o_1), \qquad 1 \leq i \leq N, \tag{4.82}$$

$$\text{Recursion}: \quad \alpha_{t+1}(j) = \Big(\sum_{i=1}^{N} \alpha_t(i) a_{ij}\Big) b_j(o_{t+1}), \qquad 1 \leq t \leq T-1, \quad 1 \leq j \leq N, \tag{4.83}$$

$$\text{Termination}: \quad P(\boldsymbol{o}|\lambda) = \sum_{i=1}^{N} \alpha_T(i), \qquad 1 \leq i \leq N. \tag{4.84}$$

In the first step, the starting value of the forward variable is set as the joint probability of the prior and the emission probability of time step 1. The

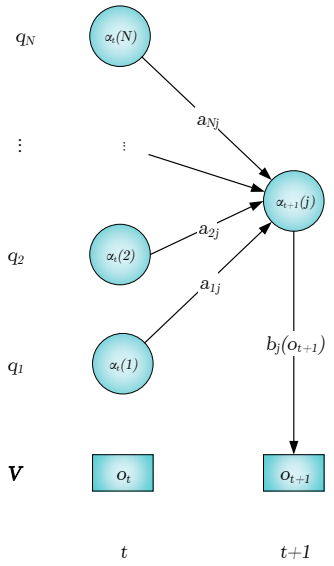


Figure 4.16.: Computation scheme for recursion of forward variable $\alpha_{t+1}(i)$.

recursion step is the propagation through the entire state space trellis, which is visualized for one of N summations in figure 4.16 For every possible state j in time step $t+1$, only every forward variable $\alpha_t(i)$ with $1 \leq i \leq N$ from the previous time step t, is summed up and joined with the emission probability of the evaluated step $b_j(o_{t+1})$. This "forward procedure" is propagated through the entire sequence until step $T-1$ is reached. Ultimately the termination step provides the desired probability $P(\boldsymbol{o}|\lambda)$ by summing up the forward variables of time step T. With the forward algorithm, the number of operations can be reduced to the order $\approx N^2T$, which is about 69 orders less than the direct calculation with $\approx 2TN^T$. This algorithm will be used later on, to determine the probabilities of a given sequence of observations, being produced by either one of several different HMMs. Ultimately, the pseudo code is presented in algorithm 6.

Algorithm 1 Solving the evaluation problem with the forward algorithm

```
1: function FORWARD-ALGORITHM(o, λ)
2:     create a forward probability matrix α
3:     for state i from 1 to N do
4:         α[i, 1] ← π_i · b_i(o_1)                          ▷ Initialization
5:     for time t from 1 to T − 1 do
6:         for state j from 1 to N do
7:             α[j, t + 1] ← Σ_{i=1}^{N} α[i, t] · a_ij · b_j(o_{t+1})   ▷ Recursion
8:     α[N, T] ← Σ_{i=1}^{N} α[i, T] · a_iN                  ▷ Termination
9:     return α[N, T]
```

The Decoding Problem

This use case deals with the determination of the "best fitting" state sequence $\boldsymbol{q}^* = (q_1^*, ..., q_T^*)$ given the model $\lambda = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi})$ and the observation sequence $\boldsymbol{o} = (o_1, ..., o_T)$. Since the definition of the "best fit" depends on the optimization criterion, more than one algorithm exist that is capable of solving the decoding problem. However, the most popular method is the so called **Viterbi algorithm**, that aims to maximize the probability of a state sequence, given the observation sequence and the model $P(\boldsymbol{q}|\boldsymbol{o}, \lambda)$. Its objective is to find one single path, that is considered to be the optimum. By applying equation 4.77, the maximization can be rewritten as follows [20, 6]:

$$\begin{aligned}\max_{q_1,q_2,...,q_{t-1}} P(\boldsymbol{q}|\boldsymbol{o}, \lambda) &= \max_{q_1,q_2,...,q_{t-1}} \frac{P(\boldsymbol{o}, \boldsymbol{q}|\lambda)}{P(\boldsymbol{o}|\lambda)} \\ &= \max_{q_1,q_2,...,q_{t-1}} P(\boldsymbol{o}, \boldsymbol{q}|\lambda) \\ &= P(\boldsymbol{o}, \boldsymbol{q}^*|\lambda) = P^*(\boldsymbol{o}|\lambda) \quad . \end{aligned} \tag{4.85}$$

For maximization, the constant value of $P(\boldsymbol{o}|\lambda)$ does not matter, which allows the above relation. To formulate the procedure properly, an auxiliary quantity is introduced, that respects equation 4.85, i.e.:

$$\begin{aligned}\delta_t(i) &= \max_{q_1,q_2...,q_{t-1}} P(q_1, q_2, ..., q_t = S_i, o_1, o_2..., o_t|\lambda) \\ &= \max_{q_1,q_2...,q_{t-1}} P(o_1, o_2..., o_t, q_1, q_2..., q_t = S_i|\lambda) \quad . \end{aligned} \tag{4.86}$$

This quantity represents the probabilities for the optimal path through the state space trellis, that produced the observation sequence until time t and can be solved recursively. The main objective however, is to find the optimal state sequence $\boldsymbol{q}^*$, which is why a tracking of the argument i that maximizes the probability $\delta_t(i)$ is necessary. Therefore the array $\psi_t(i)$ is introduced, which holds the information about the state corresponding to the maximum probability. The procedure is similar to the forward algorithm, only with a maximization over all possible state probabilities, instead of a summation and an additional operation that fills $\psi_t(i)$ [20, 6, 23]. Figure 4.16 can be considered, only with $\delta_t(i)$ instead of $\alpha_t(i)$.

$$\text{Initialization :} \quad \delta_1(i) = \pi_i b_i(o_1), \qquad 1 \leq i \leq N, \tag{4.87}$$

$$\psi_1(i) = \mathbf{0}, \tag{4.88}$$

$$\text{Recursion :} \quad \delta_{t+1}(j) = \Big(\max_i \delta_t(i) a_{ij}\Big) b_j(o_{t+1}), \qquad 1 \leq t \leq T-1,$$

$$1 \leq j \leq N, \tag{4.89}$$

$$\psi_{t+1}(j) = \operatorname*{argmax}_{1 \leq i \leq N} \Big(\delta_t(i) a_{ij}\Big), \qquad 1 \leq t \leq T-1,$$

$$1 \leq j \leq N, \tag{4.90}$$

$$\text{Termination :} \quad P^*(\boldsymbol{o}|\lambda) = P(\boldsymbol{o}, \boldsymbol{q}^*|\lambda) = \max_{1 \leq i \leq N} \delta_T(i), \tag{4.91}$$

$$q_T^* = \operatorname*{argmax}_{1 \leq i \leq N} \delta_T(i), \tag{4.92}$$

$$\text{Backtracking :} \quad q_t^* = \psi_{t+1}(q_{t+1}^*), \qquad T-1 \geq t \geq 1. \tag{4.93}$$

In contrast to the forward algorithm which computes the total output probability, this procedure only processes the most probable possibilities by maximizing over all distinct states N in time step t. The backtracking step ultimately delivers the "best fit" for the state sequence $\boldsymbol{q}^*$.

Assuming a model $\lambda = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi})$ and an observation sequence from time step 1 to t, $\boldsymbol{o} = (o_1, o_2, ..., o_t)$ is given, the Viterbi algorithm can compute the optimal state sequence. Hence, the last state of the obtained sequence $\boldsymbol{q}^* = (q_1^*, q_2^*, ..., q_t^*)$ is the currently occupied state. The Viterbi algorithm fulfills the first requirement of prognostics, which is to determine the current HS. Algorithm 7 provides the pseudo code, to solve die decoding problem.

Algorithm 2 Solving the decoding problem with the Viterbi algorithm

function VITERBI-ALGORITHM($\boldsymbol{o}$, λ)
  create a path probability matrix δ
  create backtracking matrix ψ
  create most likely state path q^*
  for state i from 1 to N **do**
    $\delta[i,1] \leftarrow \pi_i \cdot b_i(o_1)$ ▷ Initialization
    $\psi[i,1] \leftarrow 0$
  for time t from 1 to $T-1$ **do**
    for state j from 1 to N **do**
      $\delta[j,t+1] \leftarrow \max_{i=1}^{N} \delta[i,t] \cdot a_{ij} \cdot b_j(o_{t+1})]$ ▷ Recursion
      $\psi[j,t+1] \leftarrow \text{argmax}_{i=1}^{N} \delta[i,t] \cdot a_{ij}$
  $\delta[N,T] \leftarrow \max_{i=1}^{N} \delta[i,T] \cdot a_{iN}$ ▷ Termination
  $\psi[N,T] \leftarrow argmax_{i=1}^{N} \delta[i,T] \cdot a_{iN}$
  for time t from $T-1$ to 1 **do**
    $q^*[1,t] \leftarrow \psi[q^*[t+1], t+1]$ ▷ Backtracking
  return q^*

The Learning Problem

A method is presented, that aims to estimate the model parameters $\boldsymbol{A}, \boldsymbol{B}$, and $\boldsymbol{\pi}$, given an observations sequence $\boldsymbol{q}$, by maximizing $P(\boldsymbol{o}|\lambda)$ locally. In other words, the model parameters are tuned in order to find the HMM that most likely produced the given observation sequence. There exist several different numeric approaches to tackle that problem, which all belong to the family of **EM** (expectation maximization or expectaion modification) algorithms. Within this thesis, the wide spread **Baum-Welch Algorithm** is presented, which is an iterative procedure, that numerically solves for the system parameters, step by step. Initial "guesses" of a_{ij}, $b_j(k)$ and π_i are necessary to run the algorithm, which re-estimates them as $\hat{a}_{ij}$, $\hat{b}_j(k)$ and $\hat{\pi}_i$. In order to pin down the mathematics behind the Baum-Welch algorithm, further quantities have to be defined. First of all, the auxiliary backward variable, which is similar to the forward variable, is defined as follows [20]:

$$\beta_t(i) = P(o_{t+1}, o_{t+2}, ..., o_T | q_t = S_i, \lambda) \quad . \tag{4.94}$$

Whereas the forward variable considers the observation sequence from step $t = 1$ to the current step t, the backward variable accounts for the partial sequence from current step t until the end of the sequence T. Beta can be solved recursively, by initialization at time step T and termination at $t = 1$ [20, 6]:

$$\text{Initialization :} \qquad \beta_T(i) = 1, \qquad 1 \leq i \leq N, \tag{4.95}$$

$$\text{Recursion :} \qquad \beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(i), \qquad T-1 \geq t \geq 1,$$

$$1 \leq i \leq N, \tag{4.96}$$

$$\text{Termination :} \qquad P(\boldsymbol{o}|\lambda) = \sum_{i=1}^{N} \pi_i b_i(o_i) \beta_1(i), \qquad 1 \leq i \leq N. \tag{4.97}$$

Figure 4.17 illustrates the recursion step of the solution. The backward variable represents the probability that the model has been in state S_i at time t, accounting all possible transitions a_{ij} to state S_j, and the emitted observation $b_j(o_{t+1})$. Just like a mirrored forward probability, $\beta_t(i)$ is calculated via summing over all possible states N at the time step $t + 1$.

Another quantity of note, is the state occupation probability, or posterior probability of being at state S_i at time t, which can be written as

$$\gamma_t(i) = P(q_t = S_i|\boldsymbol{o}, \lambda) \quad . \tag{4.98}$$

Unlike the forward an backward probabilities, which account only for partial sequences of observations, namely from $t = 1$ to t, and from t onward to T respectively, the state occupation probability considers the entire observation sequence $\boldsymbol{o}$. Applying Bayes' rule from equation 4.79, the desired probability can be written as [6]:

$$P(q_t = S_i|\boldsymbol{o}, \lambda) = \frac{P(q_t = S_i, \boldsymbol{o}|\lambda)}{P(\boldsymbol{o}|\lambda)} \quad . \tag{4.99}$$

The denominator can be seen as normalization factor, hence $\sum_{i=1}^{N} \gamma_t(i) = 1$. The numerator can be split into two partial probabilities, namely $P(o_1, o_2, ..., o_t, q_t =$

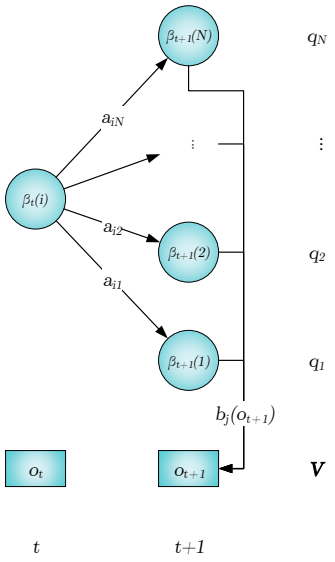


Figure 4.17.: Computation scheme for recursion of backward variable $\beta_t(i)$.

$S_i|\lambda)$ and $P(o_{t+1}, o_{t+2}, ..., o_T, q_t = S_i|\lambda)$, which leads to the following representation of the state occupation probability [6].

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P(\boldsymbol{o}|\lambda)} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)} \tag{4.100}$$

The combination of the forward and backward variable is referred to as the **forward-backward algorithm**, which is used to compute the probability of being in state S_i at time t.

Before the iterative re-estimation procedure can be executed, yet another quantity is necessary to be defined. $\zeta_t(i,j)$ is the probability of the model, being at state S_i at time t and at state S_j at time $t+1$, given the sequence and the model, i.e.:

$$\zeta_t(i,j) = P(q_t = S_i, q_{t+1} = S_j|\boldsymbol{o}, \lambda) \quad . \tag{4.101}$$

Figure 4.18 illustrates the computation scheme for $\zeta_t(i,j)$. Assuming time step t as the current one, a transition from S_i to S_j can be interpreted as the joint event of the forward probability, accounting for the observation sequence

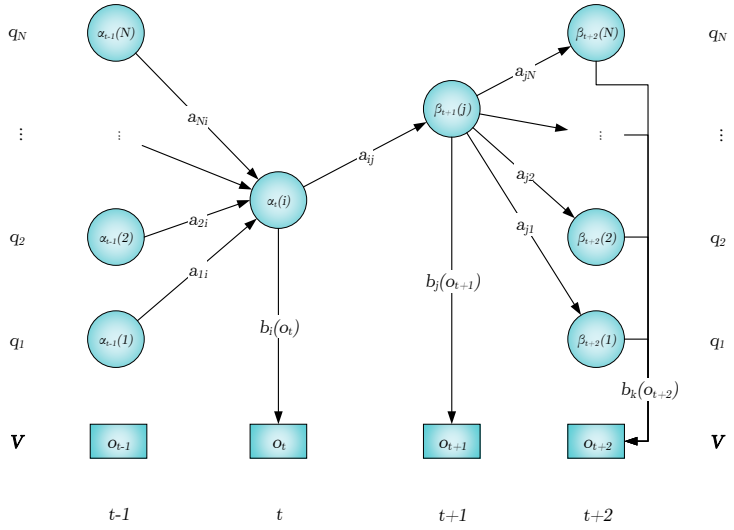


Figure 4.18.: Computation scheme for joint probability $\zeta t(i,j)$ of the model, being in S_i at t and in S_j at $t+1$ [20].

earlier to t, the backward probability, accounting for the observation sequence onward to T and ultimately the transition from S_i to S_j itself, accounting the transition probability a_{ij} and the produced emission $b_j(o_{t+1})$. $\zeta_t(i,j)$ is normalized by the total output probability $P(\boldsymbol{o}|\lambda)$, which has to be evaluated for the path, that includes the defined transition from S_i to S_j, i.e.:

$$\begin{aligned}\zeta_t(i,j) &= \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{P(\boldsymbol{o}|\lambda)} \\ &= \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)} \quad . \end{aligned} \tag{4.102}$$

Both above defined auxiliary quantities are tensors, where $\gamma_t(i)$ is two-dimensional of size $N \times T$ and $\zeta_t(i,j)$ is three-dimensional of size $N \times N \times T$. Excluding the termination step T, a summation over time can be interpreted

as the expected number of state transitions which yields [20]:

$$\sum_{t=1}^{T-1} \gamma_t(i) = \text{Number of transitions from } S_i \quad ,$$
$$\sum_{t=1}^{T-1} \zeta_t(i,j) = \text{Number of transitions from } S_i \text{ to } S_j \quad .$$

With the above findings, the re-estimated system parameters can be formulated as follows. A verbal representation of the equations is added, in order to facilitate the idea behind them [20]:

$$\begin{aligned} \hat{\pi}_i &= \text{Probability of being in state } S_i \text{ at time } (t=1) \\ &= \gamma_1(i) \quad , \end{aligned} \tag{4.103}$$

$$\begin{aligned} \hat{a}_{ij} &= \frac{\text{Number of transitions from } S_i \text{ to } S_j}{\text{Number of transitions from } S_i} \\ &= \frac{\sum_{t=1}^{T-1} \zeta_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \quad , \end{aligned} \tag{4.104}$$

$$\begin{aligned} \hat{b}_j(k) &= \frac{\text{Total visits in } S_j \text{ when observing } V_k}{\text{Total visits in } S_j} \\ &= \frac{\sum_{\substack{t=1 \\ t:o_t=V_k}}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} \quad . \end{aligned} \tag{4.105}$$

Where the second argument of the summation in the numerator of equation 4.105, restricts the relevant terms that contribute to the re-estimated emission probability, to the events, where the considered observation k actually occurred. In contrast to the transition probability $\hat{a}_{ij}$ that considers transitions between states, the re-estimate $\hat{\pi}_i$ and $\hat{b}_j(k)$ account for visits in the respective states, which is why the summation includes the termination step T. For the prior probability $\hat{\pi}_i$, the visits in states S_i can be computed without summation because it is considered to be in the first time step $t = 1$. During the course of the re-estimation procedure, the stochastic constraints of the parameters are

preserved, i.e,

$$\sum_{i=1}^{N} \hat{\pi}_i = 1 \quad , \tag{4.106}$$

$$\sum_{j=1}^{N} \hat{a}_{ij} = 1 \quad , \qquad 1 \leq i \leq N, \tag{4.107}$$

$$\sum_{k=1}^{K} \hat{b}_j(k) = 1 \quad , \qquad 1 \leq j \leq N. \tag{4.108}$$

By denoting the re-estimated HMM as $\hat{\lambda} = (\hat{\boldsymbol{A}}, \hat{\boldsymbol{B}}, \hat{\boldsymbol{\pi}})$ and the initial version as $\lambda = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi})$, Baum *et.al.* proved that the observation sequence $\boldsymbol{o}$ is either equally, or more likely to be produced by the re-estimated HMM $\hat{\lambda}$. They achieved that by maximizing the auxiliary function $Q(\lambda, \hat{\lambda})$ [20].

$$Q(\lambda, \hat{\lambda}) = \sum_{Q} P(\boldsymbol{q}|\boldsymbol{o}, \lambda) \log\left(P(\boldsymbol{o}, \boldsymbol{q}|\hat{\lambda})\right) \tag{4.109}$$

$$\max_{\hat{\lambda}} Q(\lambda, \hat{\lambda}) \quad \Longrightarrow \quad P(\boldsymbol{o}|\hat{\lambda}) \geq P(\boldsymbol{o}|\lambda) \tag{4.110}$$

with respect to equations 4.106 to 4.108. Javier R. Movellan [19] presents a well structured derivation of this proof. Although the procedure delivers a better solution with every iteration, it is in fact just a local maximum that is found, which stresses the necessity of a rather good first guess of λ.

Within this thesis, the EM algorithm is going to be used to train proper HMMs. As for now, the required input is a training data set $\boldsymbol{o} = (o_1, o_2, ..., o_T)$ and, if available, a proper guess of the initial model parameters. If the latter is a only a rough guess, the algorithm will certainly work, however the result will be a local maximum, whose quality depends on the shape of the maximization surface. The pseudo code is represents in algorithm 8.

4.3.2. Extension of the Left-Right Hidden Markov Model

As mentioned in the previous section, a rather simple type of HMM is used to model the deterioration of mechanical components, precisely the degradation of the secondary vertical damper. That is a left-right HMM, which is sometimes

Algorithm 3 Solving the learning problem with the EM algorithm

function EM-ALGORITHM($\boldsymbol{o}$, a_{est}, b_{est}, π_{est}, $Iter$)
  for iterations $iter$ from 1 to $Iter$ **do**
    create probability arrays α, β, γ, ζ
    if $iter \leq 1$ **then**
      $a \leftarrow a_{est} \quad b \leftarrow b_{est} \quad \pi \leftarrow \pi_{est}$
    else
      $a \leftarrow \hat{a} \quad b \leftarrow \hat{b} \quad \pi \leftarrow \hat{\pi}$
    for state i from 1 to N **do**
      $\alpha[i,1] \leftarrow \pi_i \cdot b_i(o_1)$
      $\beta[i,1] \leftarrow 1$ ▷ Initialization
    for time t from 1 to $T-1$ **do**
      for state j from 1 to N **do**
        $\alpha[j,t+1] \leftarrow \sum_{i=1}^{N} \alpha[i,t] \cdot a_{ij} \cdot b_j(o_{t+1})]$ ▷ Forward recursion
    for time t from $T-1$ to 1 **do**
      for state i from 1 to N **do**
        $\beta[i,t] \leftarrow \sum_{j=1}^{N} a_{ij} \cdot b_j(o_t) \cdot \beta[j,t+1]$ ▷ Backward recursion
    for time t from 1 to T **do**
      for state j from 1 to N **do**
        $\gamma[j,t] \leftarrow \frac{\alpha[j,t] \cdot \beta[j,t]}{\sum_{i=1}^{N} \alpha[i,t] \cdot \beta[i,t]}$
    for time t from 1 to $T-1$ **do**
      for state i from 1 to N **do**
        for state j from 1 to N **do**
          $\zeta[i,j,t] \leftarrow \frac{\alpha[i,t] \cdot a_{ij} \cdot b_j(o_{t+1})] \cdot \beta[j,t]}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha[i,t] \cdot a_{ij} \cdot b_j(o_{t+1})] \cdot \beta[j,t]}$
    for state i from 1 to N **do**
      $\hat{\pi}_i \leftarrow \gamma[i,1]$
      for state j from 1 to N **do**
        $\hat{a}_{i,j} \leftarrow \frac{\sum_{t=1}^{T-1} \zeta[i,j,t]}{\sum_{t=1}^{T-1} \gamma[i,t]}$
    for state j from 1 to N **do**
      for observation k from 1 to K **do**
        $validObs \leftarrow (\boldsymbol{o} == V_k)$
        $\hat{b}_j(V_k) \leftarrow \frac{\sum_{t=1}^{T} (\gamma[j,t] \circ validObs)}{\sum_{t=1}^{T} \gamma[j,t]}$
  return $\lambda \leftarrow (\hat{\pi}, \hat{a}, \hat{b})$

referred to as "Bakis HMM" in literature. Yet another constraint for the state transitions of the left-right model is, that no state can be skipped, or in other words, the state transition is only possible to the immediate successor state. Nevertheless, deterioration can be modeled pretty well, because the unidirectional, probabilistic propagation through the state-time trellis, is able to represent any possible non-linearity. However, the HMM as a detection and prediction tool, will only work properly, if the observed sequence is somewhat similar to the sequence used for training. Since that can hardly be guaranteed by a single HMM, Le *et.al.* [16] introduced the concept of a so called multi-branch MB-HMM. The idea is, to train separate HMMs for every distinct deterioration mode that can occur. This enables the modeling of competing failure modes, which may or may not occur at the same time [16]. Figure 4.19 illustrates the coexistence of several branches. Assuming a MB-HMM with

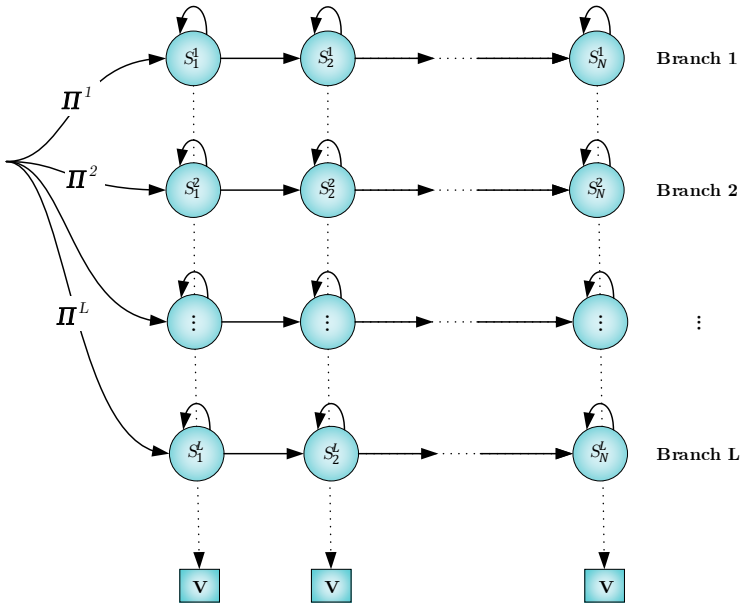


Figure 4.19.: Multiple branch, left-right automaton for modeling of coexistent failure modes.

two branches, one for rapid, the other for slow deterioration. An observed sequence $\boldsymbol{o}$ will initially be classified in terms of its likeliest branch. The prior probability Π determines the initial likelihood of every branch, enabelign an unequal distribution of the different branches. Thereafter, the prediction of the RUL will be performed through Bayes' averaging with the constituent models' likelihood as weights. An inter-branch state transition is not possible, however every branch contributes to the final result as follows[16]. Considering equations 4.79 and 4.78, the model likelihood can be written as

$$P(\lambda_l|\boldsymbol{o}) = \frac{P(\boldsymbol{o}|\lambda_l)P(\lambda_l)}{\sum_{l=1}^{L}\Big(P(\boldsymbol{o}|\lambda_l)P(\lambda_l)\Big)} \quad . \tag{4.111}$$

Where $L \in \mathbb{N}$ is the total number of coexisting branches. The probability of the observation sequence, given the model $P(\boldsymbol{o}|\lambda_l)$ is determined for every branch with the forward procedure, described in section 4.3.1. The prior probability $P(\lambda_l)$, is chosen according to the respective deterioration modes' event risk. Every branch will contribute to the final result, yet the best fitting model will have the biggest share, considering its relatively higher likelihood. By providing a representative selection of branches (failure modes), a wide range of possible observation sequences can be predicted properly. The impact of number and parameters of the distinct HMMs (branches), as well as the determination of their likelihood is rather big, which is why at this point an expert consultation is advisable. Moreover, the prior probability $P(\lambda_l)$ has to be chosen based on experience. The actual RUL calculation will be discussed in the next chapter, nonetheless its appearance in the Bayesian model averaging is presented hereafter [10], i.e.:

$$\mathbb{E}(RUL|\boldsymbol{o}) = \sum_{l=1}^{L}\mathbb{E}(RUL|\lambda_l,\boldsymbol{o})P(\lambda_l|\boldsymbol{o}) \quad . \tag{4.112}$$

Where $\mathbb{E}(\cdot)$ denotes the expected value, which is in this case the overall RUL given the observation sequence on the left-hand side and the branch specific RUL, given the model and the observations sequence, on the right-hand side. The weight for the averaging is $P(\lambda_l|\boldsymbol{o})$, which is calculated with equation 4.111. An implementation of Bayesian model avaraging is presented in algorithm 9.

Algorithm 4 MB-HMM contribution with Bayesian model averaging

1: **function** BAYESIANMODELAVERAGING($P(\boldsymbol{o}_{test}|\lambda)$, $P(\lambda)$,RUL_{ψ}^{l})
2: **for** branch l from 1 to L **do**
3: $P(\lambda_l|\boldsymbol{o}_{test}) \leftarrow \frac{P(\boldsymbol{o}_{test}|\lambda_l)\cdot P(\lambda_l)}{\sum_{l=1}^{L} P(\boldsymbol{o}_{test}|\lambda_l)\cdot P(\lambda_l)}$
4: $RUL_{\psi} \leftarrow \sum_{l=1}^{L} RUL_{\psi}^{l} \cdot P(\lambda|\boldsymbol{o}_{test})$
5: **return** RUL

4.3.3. Solution for the Remaining Useful Lifetime Algorithm

In sections 4.3.1 and 4.3.2, the tools were defined, that are necessary to develop an algorithm, capable of fulfilling both prognostics requirements, namely the HS detection and the RUL prediction. Referring to figure 3.4, the HMM related actions take place in the second and third stage, involving training and testing. This section presents the complete framework for both of those actions, together with the used tools and methods. The input data is described earlier in chapter 4. Figure 4.20 shows the complete procedure from input data to a RUL statement and points out where the algorithms defined in section 4.3.1 are applied.

Learning and Testing Starting with the training of HMMs, the EM algorithm (or Baum-Welch) is the first one to be applied, solving the "learning problem" according to section 4.3.1. It is performed for every data set corresponding to a different branch, resulting in a set of HMMs $\boldsymbol{\lambda}_l = (\lambda_1, \lambda_2, ..., \lambda_L)$, each modeling a different deterioration mode. Together they represent the MB-HMM, which then can be tested with a new, unknown and independent data set. In doing so, the so called out-of-sample error can be determined, which is basically a quantity that remains unknown. Usually, only the training data is labeled, providing the desired output of the model. Testing the MB-HMM means, processing of measured observations, which in this case are particularly derived features, classified by a SVM. Therefore the label of the data has to be determined by the model and can not be validated immediately, hence the uncertainty of the out-of-sample error. However, if the testing is performed with simulated data, the actual desired output of the HMM is known, because the components are conditioned in course of the multi-body

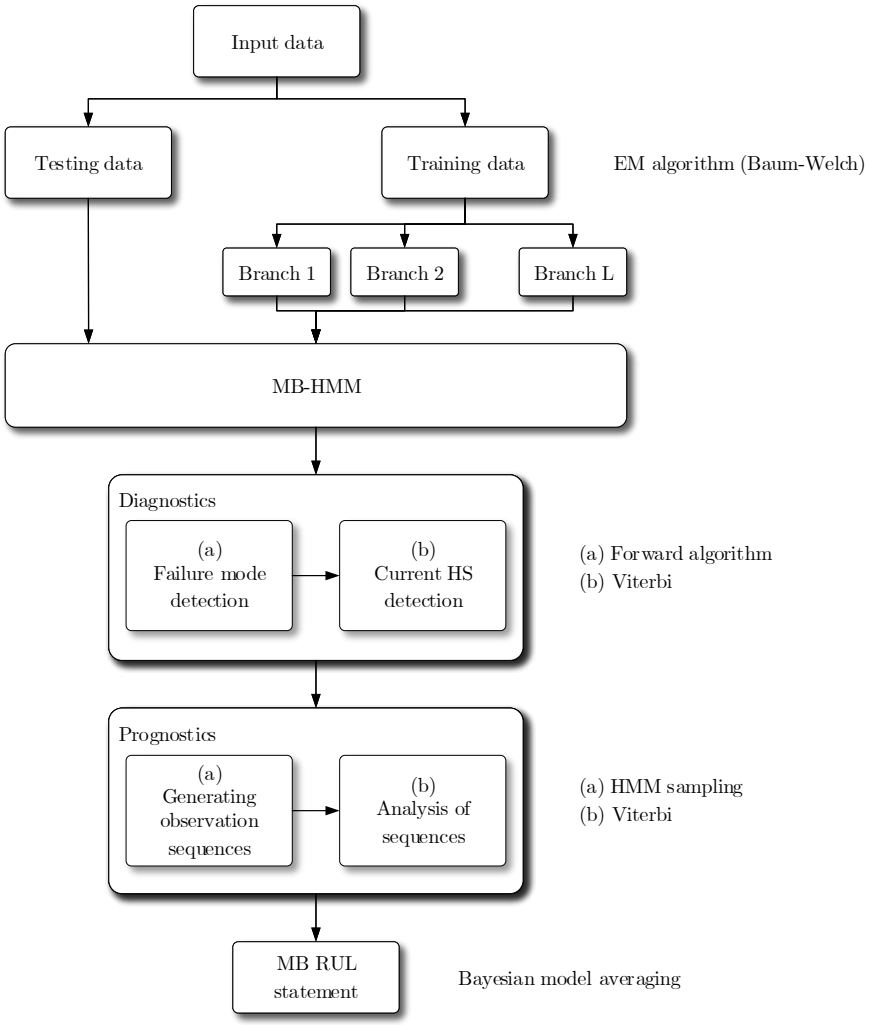


Figure 4.20.: Framework for RUL estimation with MB-HMM.

simulation. Therefore, the following diagnosis and prognosis step can be assessed, which will happen at the end of the procedure by comparison of the "real" output with the estimated output.

Diagnostics Within the diagnostics block, two algorithms are applied, namely the forward algorithm and the Viterbi algorithm. Since the MB-HMM consists of different branches, the first step is to determine the likelihood of each constituent HMM, according to the procedure in section 4.3.2, which is called the "evaluation problem". The output is a probability measure that classifies the applied testing data set in terms of its relation to one of the distinct branches. After successfully applying the algorithm, the contribution of each branch to the ultimate output is fixed. Moving on to the next step, the Viterbi algorithm determines the current HS of each constituent branch, as well as the most likely state path from $t = 1$ until the current time step t. The exact procedure is provided in section 4.3.1 and is also referred to as the "decoding problem".

Prognostics This block deals with the estimation of the future evolution of the HS, which can be seen, in terms of HMMs, as the ongoing propagation through the state time trellis until the last state S_N is reached. Once the current state S_{cs} is determined by the Viterbi algorithm, the corresponding time t can be evaluated by counting the number of time steps of the Viterbi sequence for each state Δt_i for $i = 1, ..., cs$. From there on out, the prognosis estimates the future state transitions and counts the time steps Δt_i for $i = cs, ..., N$, which are required until the last state is reached. To achieve this, a statistically representative number of observation sequences Y is produced with all branches L and with respect to the constituent HMM parameters λ_l. Based on the characteristics of those sequences, a statistical statement about the mean and the standard derivation of the RUL can be made. The generation of $\boldsymbol{o}_y^l = (\boldsymbol{o}_1^1, \boldsymbol{o}_2^1, ..., \boldsymbol{o}_Y^1; \boldsymbol{o}_1^L, \boldsymbol{o}_2^L, ..., \boldsymbol{o}_Y^L)$ sequences, can be described in four steps [20], which are basically called, sampling an HMM:

1. Evaluate the initial state through the prior probability π and set the time $t = 1$.
2. Select an observation $o_t = V_k$ according to state dependent emission probability $b_j(k)$, given by the emission matrix $\boldsymbol{B}$. Figure 4.21 illustrates

the different probability mass functions (PMFs) for the respective state. Depending on which state the model is in, an observation V_k is picked with respect to its probability.

3. Go on to the next time step $t+1$ and select the new state q_{t+1} according to the state transition probability a_{ij} provided by $\boldsymbol{A}$.
4. Repeat step 2) to 4) with $t = t+1$ until $q_{t+1} = S_N$.

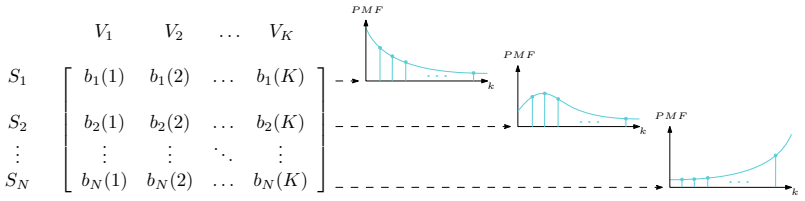


Figure 4.21.: PDF for different states (rows) within the emissions matrix $\boldsymbol{B}$.

After repeating the above procedure Y times for each branch L, the produced sequences are analyzed, again by the Viterbi algorithm, producing the likeliest state sequences $\boldsymbol{q}^*_{y,l} = (\boldsymbol{q}^*_{1,1}, \boldsymbol{q}^*_{2,1}, ..., \boldsymbol{q}^*_{Y,1}; \boldsymbol{q}^*_{1,L}, \boldsymbol{q}^*_{2,L}, ..., \boldsymbol{q}^*_{Y,L})$. The Viterbi sequence can slightly differ from the generated sequence, obtained by Rabiner's procedure, since Viterbi always takes the most likely way through the model. Thereafter, these generated Viterbi sequences are analyzed in terms of their required steps $\Delta t^l_{i,y}$ per state i, from the previously determined, current state $S_i = S_{cs}$, until the last state S_N. l denotes the branch and y the generated sequence. The number of time steps within the current state are considered as well, i.e. the predicted number of steps for S_{cs} is decreased by the previously determined amount. This detail is shown in algorithm 10, however the theoretical mathematics consider $\Delta t^l_{i,y}$ to count exactly from the current step, until the transition to S_N. This quantity is also known as the RUL. A separate statistical evaluation of all sequences from the different branches e.g. $\boldsymbol{q}^*_{y=1,l=1}(t) = (q^*_{1,1}(t=1), q^*_{1,1}(t=2), ..., q^*_{1,1}(t=T)),\ \forall y$, delivers an expected value RUL^l_μ as well as an upper and a lower bound for each branch. These statistical quantities are evaluated for each state separately, which yields in the RUL per branch and state as $RUL^l_{\mu,i}$. This applies to the bounds as well, which are defined as the standard derivation $\mathbb{S}(\cdot)$ of the above evaluation,

i.e.:

$$RUL^{l}_{\mu,i} = \mathbb{E}\left(\Delta t^{l}_{i,y}\right), \qquad 1 \leq y \leq Y \qquad \forall i,l \tag{4.113}$$

$$RUL^{l}_{upper,i} = \mathbb{E}\left(\Delta t^{l}_{i,y}\right) + \mathbb{S}\left(\Delta t^{l}_{i,y}\right), \qquad 1 \leq y \leq Y \qquad \forall i,l \tag{4.114}$$

$$RUL^{l}_{lower,i} = \mathbb{E}\left(\Delta t^{l}_{i,y}\right) - \mathbb{S}\left(\Delta t^{l}_{i,y}\right), \qquad 1 \leq y \leq Y \qquad \forall i,l \tag{4.115}$$

By executing the above equations for all states and branches, the obtained result in a matrix for each of the three quantities, where the rows correspond to the branches and the columns to the states of the MB-HMM. Each row contains the estimated number of steps per state, or the upper and lower bound respectively. Hence, those matrices are of size $L \times N$ and index ψ is either μ, $upper$ or $lower$, as follows:

$$\boldsymbol{R\hat{U}L}_{\psi} = \begin{bmatrix} RUL^{1}_{\psi,1} & RUL^{1}_{\psi,2} & \dots & RUL^{1}_{\psi,N} \\ RUL^{2}_{\psi,1} & RUL^{2}_{\psi,2} & \dots & RUL^{2}_{\psi,N} \\ \vdots & \vdots & \ddots & \vdots \\ RUL^{L}_{\psi,1} & RUL^{L}_{\psi,2} & \dots & RUL^{L}_{\psi,N} \end{bmatrix} \quad . \tag{4.116}$$

Thereafter, the RUL statement with the contribution from every branch can be computed by applying Bayesian model averaging from equation 4.112. This operation results in a row vector of size $1 \times N$, providing the time steps, or RUL for each state as

$$\boldsymbol{RUL}_{\psi} = \begin{bmatrix} RUL_{\psi,1} & RUL_{\psi,2} & \dots & RUL_{\psi,N} \end{bmatrix} \quad , \tag{4.117}$$

where the index ψ is again either μ, $upper$ or $lower$, representing the mean or the respective confidence bounds. Given the scenario that the Viterbi algorithm would detect the third state as current state when analyzing an arbitrary test sequence, the above described procedure would just deliver 0 as results for the first two states and some $RUL > 0$ for the remaining ones. To obtain a quantity for the overall RUL, the above equation 4.117 is summed up, i.e.:

$$RUL_{\psi} = \sum_{i=1}^{N} RUL_{\psi,i} \quad . \tag{4.118}$$

The pseudo code in algorithm 10 is supposed to give a better understanding of the framework. Among the above defined algorithms 6 to 9, a sampling

function from the Matlab library, called "hmmgenerate", is used for point (a) from the prognostics block in figure 4.20. It should be kept in mind, that the number of branches L, number of states N, number of observations K and length of the sequences T are considered to be known or chosen at this point.

Algorithm 5 RUL

procedure RUL-ESTIMATE($\boldsymbol{o}_{train}$,$\boldsymbol{o}_{test}$,a_{est},b_{est},π_{est},$Iter$)
  for branch l from 1 to L **do**
    function EM-ALGORITHM($\boldsymbol{o}^{l}_{train}$,$a_{est}$,$b_{est}$,$\pi_{est}$,$Iter$)
      return $\lambda_l \leftarrow (\hat{\pi},\hat{a},\hat{b})$
    function FORWARD-ALGORITHM($\boldsymbol{o}_{test}$, λ_l)
      return $P(\boldsymbol{o}_{test}|\lambda_l)$
    function VITERBI-ALGORITHM($\boldsymbol{o}_{test}$, λ_l)
      return $\boldsymbol{q}^*$
    $currentState \leftarrow q^*[end]$
    $numCurrentState \leftarrow length(find(\boldsymbol{q}^* == currentState))$
    $numPreviousStates \leftarrow length(\boldsymbol{q}^*)$
    function HMMGENERATE(λ_l)
      return $\boldsymbol{o}_{gen}$]
    function VITERBI-ALGORITHM($\boldsymbol{o}_{gen}$, λ_l)
      return $\boldsymbol{q}^*$
    create time step matrix Δt
    create RUL matrices $\boldsymbol{RUL}_{\mu}$,$\boldsymbol{RUL}_{upper}$,$\boldsymbol{RUL}_{lower}$
    for gen y from 1 to Y **do**
      for state $ifind$ from 1 to N **do**
        $\Delta t[y, ifind] \leftarrow find(\boldsymbol{q}^*[y] == ifind,$ last sample$)$
    $\Delta t \leftarrow \Delta t[:, currentState : N-1]$
    $\Delta t[:, currentState] - numPreviousStates$
    for j from 1 to N **do**
      $\boldsymbol{RUL}^{l}_{\mu}[j] \leftarrow mean(\Delta t[:, j])$
      $\boldsymbol{RUL}^{l}_{upper}[j] \leftarrow mean(\Delta t[:, j] + std(\Delta t[:, j])$
      $\boldsymbol{RUL}^{l}_{lower}[j] \leftarrow mean(\Delta t[:, j] - std(\Delta t[:, j])$
  function BAYESIANMODELAVERAGING($P(\boldsymbol{o}_{test}|\lambda_l)$,$P(\lambda_l)$,$\boldsymbol{RUL}^{l}_{\mu}$,$\boldsymbol{RUL}^{l}_{upper}$,$\boldsymbol{RUL}^{l}_{lower}$)
    return $\boldsymbol{RUL}_{\mu}$,$\boldsymbol{RUL}_{upper}$,$\boldsymbol{RUL}_{lower}$
  $RUL_{\mu} \leftarrow \sum_{j=1}^{N} RUL_{\mu}[j]$
  $RUL_{upper} \leftarrow \sum_{j=1}^{N} RUL_{upper}[j]$
  $RUL_{lower} \leftarrow \sum_{j=1}^{N} RUL_{lower}[j]$
  return RUL_{μ}, RUL_{upper}, RUL_{lower}

5. Results

A lot of different tools and methods for condition prediction of mechanical components of railway vehicles have been presented within this thesis. As already mentioned in chapter 3, the data driven approach for prognosis concepts can be applied to most of the vehicles components. Nevertheless, the input data as well as the parameter selection of the statistical models are different, which is why the derived algorithm is going to be applied to a specific component, namely the secondary vertical damper, on the right side of the leading bogie. Figure 3.4 provides a detailed overview of what the exact course of action was. Having said, that the wealth of results through the entire procedure is extraordinary large, the presented results have to be carefully selected, in order to provide a good impression of what was actually achieved, without overloading this chapter. Thus the results are limited to the most significant findings.

- The equation of motion by a multi-body simulation
  - Frequency response analysis
  - Track curvature and related, filtered system response
- Mapping of the health condition onto acceleration signals by SVM
  - Trained classificator with three predictors, illustrated by the hyperplane for all considered filters
  - Performance of SVM for selected filter on independent data
- Prediction of the RUL by MB-HMM
  - State and observation sequences of trained HMMs for different branches
  - State and observation sequence of test data sets
  - RUL expressed as the equivalent remaining useful mileage

5.1. The Equation of Motion by a Multi-Body Simulation

Relating to the description in section 4.1, the required inputs for the multi-body simulation are vehicle and track parameter settings. The vehicle is configured as a generic commuter train, based on data sheets from SIEMENS. Basically all of the defined parameters are constant within each iteration of the simulation. However, in order to represent the deterioration of the damper, its damping rate is decreased between iterations. Eleven steps à 10% are combined with eleven different, randomly chosen revenue loads. These are drawn from a gamma distribution and cause a significant noise in the system response. For all those 121 simulations, the same track is loaded, which covers a distance of $40km$. The output is an object, that contains statistical quantities like the standard derivation and the mean, together with the simulation's meta data, like the velocity, track parameters and vehicle parameters. Most important of all, the output file contains the corresponding damper condition, which is in fact considered to be its HS. The output sample rate is 1 sample per $100m$ of track, hence for the considered track, every stored quantity is a vector of size 400×1.

A practicable way of managing the rather big collection of equations in section 4.1, is to define them as symbolic variables within Matlab, using the "symbolic math toolbox". After the variables are brought into the desired matrix form, they are converted to a stand alone function by "matlabFunction". Thereafter able to be called with a specific set of parameters, the stand alone function delivers the system matrices in the desired form. Ultimately, a linear state-space model object is created with "ss" and solved with "lsim".

Frequency Response Analysis

The common way to validate the output of a model, is to simulate with certain parameters, were the right result is known. For a railway vehicle with four axis, two bogies and a car body, this validation is performed through a frequency response analysis. The vertical transfer function $G = \frac{z_{Car\ Body}}{z_{Track}}$, that links the position of the track with the CoG of the car body, has certain qualitative

characteristics, which are commonly known. Hence, if the calculated transfer function and its knowingly true result are alike, the model is considered to deliver reasonable results. The reference transfer function is taken from Haigermoser [9], who presents its result, evaluated at excitation frequencies from 0 to $10Hz$, which can be interpreted as the vehicle, driving with increasing speed over a periodic excitation. It appears to have an extinction, or a significant reduction of the signal at 2, 6 and $10Hz$. For low frequencies $< 1Hz$ the car body follows the movement of the excitation with a slight amplification. Figure

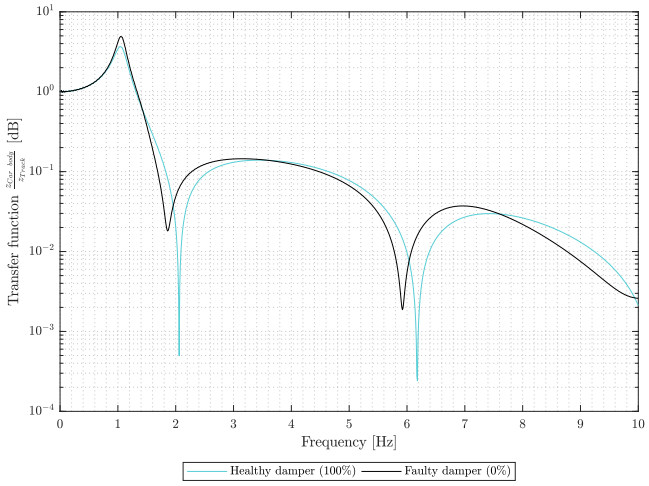


Figure 5.1.: Transfer function of the vertical positions from car body and track excitation, evaluated for frequencies from 0 to $10Hz$ with a sinusodial excitation

5.1 shows the vertical transfer function for the multi-body simulation, which is generated using a sinusodial excitation in order to create a solution, which is comparable with the reference function. The frequency range is discretized and the response for each frequency evaluated numerically. Thereafter the standard derivation of the signal, corresponding to each discrete frequency is used to derive the transfer function G. The turquoise line is the response of the model, evaluated at its nominal condition. The black line represents the corresponds to a degraded, or even broken secondary vertical damper, which is the actual quantity that is targeted by the parameter variation. Firstly, the characteristics of the transfer function are similar to the reference function,

which suggests a reasonable simulation model. Secondly, the difference of the vertical system response in case of a broken component is illustrated. However, the signal is not easy to distinguish, if the track excitation is not a perfectly smooth sinus. Therefore, the subsequently performed statistical models account more than just two feature values, that better represent any anomaly in the system response.

Track Curvature and System Response

An interesting feature of this efficient simulation, is its ability to compute lateral accelerations, induced by the track curvature. The centrifugal acceleration was simplified by means of neglecting the decreasing impact of gravity. Results show that curvature has a significant impact on the vehicle dynamics. The above simplification can be argued by the rather small rolling angles χ. The black line represents the curvature in $[m^{-}1]$, the turquoise line is the

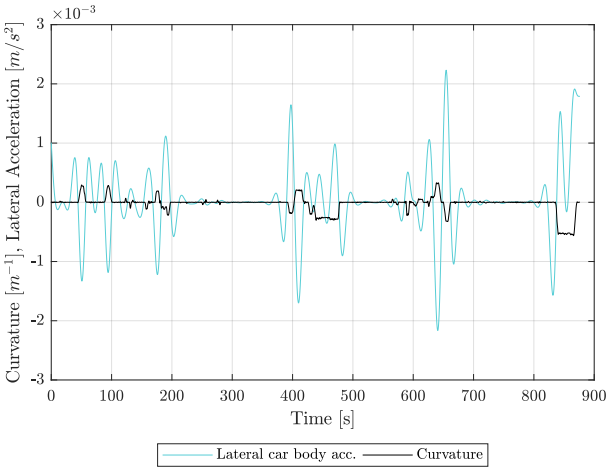


Figure 5.2.: Curvature and corresponding lateral system response with an applied low-pass filter

lateral system response of the car body in $[m/s^2]$, evaluated just above the leading bogie on the roll axis. The x-axis shows the time of one iteration of

the simulation, which again corresponds to about 40km of track. The signal was prepared with a low-pass filter, in order to see the slow response to the curve and get rid of the high frequency noise. A positive curvature indicates a right turn and results in a negative peak (to the left of the vehicle) of the lateral acceleration and vice versa. This is how the cornering was accounted within the Matlab model.

5.2. Mapping of the Health Condition onto Acceleration Signals by Support Vector Machines

The procedure described in section 4.2, forms a hyperplane within a feature space, built of a combination of three representative features. The selected number and type of features, depends on the considered component and is done by sequential feature selection according to section 3.3.2. The result of the feature selection showed, that an increased number of features (> 3) actually does deliver better results, however the improvement is marginal. This observation and the fact that a visual interpretation is required, supports the decision of only three combined features.

Trained SVM

The SVM is trained with the previously simulated and labeled (in terms of healthy and faulty) data set, with a linear kernel. A kernel transformation is not necessary, because the processed data is basically linearly separable in the input space. Nevertheless, a few outliers are part of the data, which are penalized by the SVM. Figure 5.3 shows the resulting hyperplane, that best separates the healthy (green samples) from the faulty (red samples) data points. In that case, healthy means 100% damping rate, faulty 0% respectively. Prior to the learning procedure, the complete data set is filtered with a band-pass filter of various settings, whose upper and lower frequencies are determined based on experience of the bogie analytics group at SIEMENS. For each filter setting a SVM was trained, which leads to a total of 10 SVMs for the same data set, but with varying filter settings. Figure 5.3 illustrates the selected SVM with the best generalization, hence the best classification performance

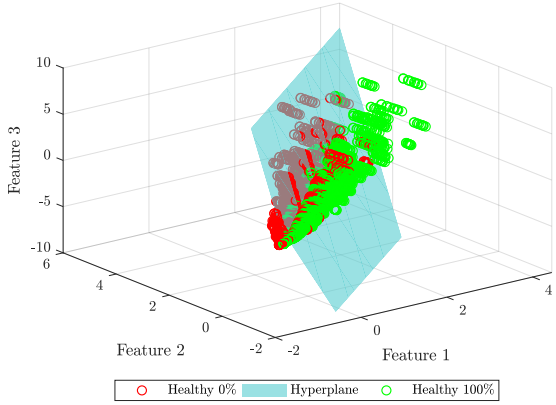


Figure 5.3.: Hyperplane of a two class SVM with three predictors (features), trained to classify a secondary vertical damper of railway vehicles

on test data. This model selection is made by a visual interpretation of the score[1] PMFs for all filter settings. Assuming the PMFs for healthy and faulty data points would overlap, it would not be possible to distinguish between classes based on the score or posterior probability, which is considered to be the output of SVMs. Whereas a distribution with two local maxima, each of them corresponding to one of the two classes, indicates that the SVM nicely generalizes. Hence, the selected filter setting corresponds to the model, with the largest distance between the two maxima of the score PMF. Figure 5.4 shows the score PMF of the selected model. The two separated maxima for healthy and faulty data points are most distinctive for a band-pass filter with a centered frequency band. This model is now used to label the intermediate conditions[2] of the damper to a corresponding score, or posterior probability measure.

[1]Quantity whose sign indicates the class (healthy, faulty) and magnitude the distance to the plane(see section 4.2)

[2]Labels between 100% and 0% damping.

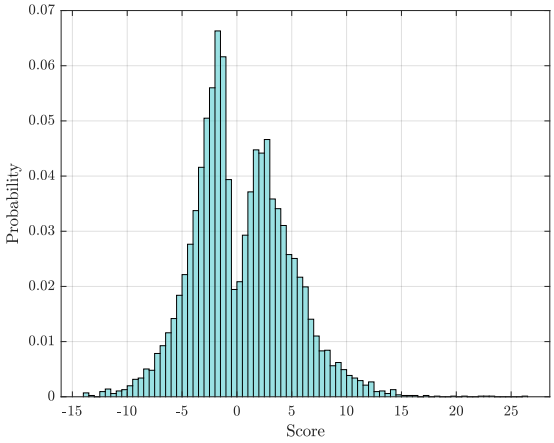


Figure 5.4.: Score PMF of the SVM with the selected filter settings (center frequency band)

Tested SVM

Following through the machine learning diagram form section 3.2, the next step is testing. Now that the hyperplane is fixed, data, corresponding to the intermediate conditions is tested, which results in a mapping of the health condition[3] to the SVMs output. According to the discussion in section 4.2.2, the output of the SVM will now be presented as the posterior probability, rather than the score. Figure 5.5 illustrates the computed output of the selected SVM, for the labeled input data. Those eleven subplots are PMFs for the posterior probability, whose value is within the interval $[0, 1]$. Obviously the model can distinguish between healthy and faulty. The posterior probability of the better part of data samples corresponding to a healthy (100% to 70%) or a faulty (0% to 30%) damper, is either 0 for healthy, or 1 for faulty. However, it has its difficulties to distinguish between the constituent healthy, average or faulty data sets, because their posterior probability outputs look very much alike. Nonetheless, a tendency is certainly identifiable, which makes the posterior probability a one-dimensional quantity that correlates to the components health condition. Each one of these eleven subplots, belongs to a sequence

[3]Health condition is determined by the parameter damping rate, which is iterative decreased throughout the multi-body simulation

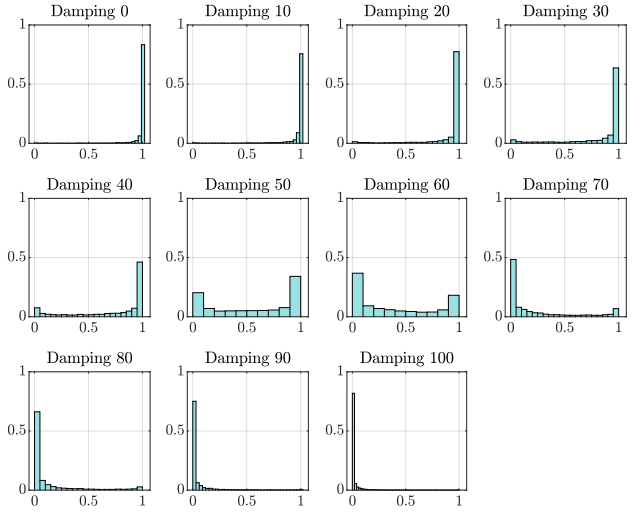


Figure 5.5.: Posterior probability computed by the selected SVM for the intermediate damper conditions.

of posterior probabilities that represents a distinct health condition. Those sequences are now concatenated according to section 4.2.3. Additionally, a sequence of the same length is produced, that consists of the HS that corresponds to the respective observation. Both of those sequences represent a labeled data set, which is used in the last phase, to train and test the HMM.

5.3. Prediction of the Remaining Useful Lifetime by Multi-Branch Hidden Markov Models

The MB-HMM consists of a certain number of distinct branches, each of them modeling a different failure mode. This approach only makes sense for rather diverse deterioration modes, therefore not every possible failure, which probably has a similar deterioration behavior, has to be modeled separately. A number of three distinct branches is chosen in order to provide one model for rapid deterioration, one for the average and another one

for slow and steady degradation. Each of those separate HMMs has seven distinct states, and ten distinct observations. As described in section 4.3, the basic HMM can only process discrete observations, i.e. the continuously provided posterior probability, has to be discretized in order to work with the algorithms. The number of states is determined by a parameter study, were the same sequences have been used for prognosis. Their results show, that to less states fail to predict the RUL, because the last state is reached way to early. Considering the damper to be broken as soon as the last state is reached, HMMs with less than seven states underestimated the RUL, because too much of the entire observation sequence corresponds to the broken state. The upper bound for the number of states is certainly hard to tell, however, too many states tend to overestimate the RUL, which is basically the more critical error. Therefore, not more than the necessary seven states were chosen. The number of distinct observations on the other hand, has a less significant impact on the models output. The reason for that, is the distribution of posterior probabilities for the constituent HS. Figure 5.5 points out, that a healthy condition produces observations somewhat close to zero, faulty condition close to one and the intermediate conditions have almost uniformly distributed observations. Hence, a more precise discretized observation sequence does not really change anything, unless it has an order of 10^3 or higher, which will certainly lead to a loss of computational performance because the system matrices are huge. Last of all, parameters for testing, which is an assessment of the model in terms of its out-of-sample error, are determined. The iterative procedure of providing partial segments of the testing sequence is described in section 3.4. The number of iterations actually determine how often the RUL is estimated throughout the lifetime. Hence, more iterations deliver a better resolution of the final result. A practicable tradeoff between computation time and resolution is reached with 20 iterations. Therefore, the synthetic test sequence is split into $\frac{1}{20}$ segments of increasing length. The relevant parameter setting for MB-HMM training and testing are summed up as follows.

- Training parameters
  - Number of branches: $L = 3$
  - Number of distinct states: $N = 7$
  - Number of distinct observations: $M = 10$
- Testing parameters

- Prior probability of the branches $\Pi_l = \frac{1}{3},\ \forall l$
- Tested synthetic sequences: $J = 3$
- Iterations for partially provided test sequences: $i_{Test} = 20$

Training of an MB-HMM

Three representative training sequences are produced, in order to model rapid, average and slow deterioration. Those three sequences, represent the branches of the MB-HMM. The assumed average mileage until the EoL is chosen based on the current maximum mileage during which a damper operates. 1.6 million kilometers are selected as average, 1.0 million as rapid and 2.1 million as slow deterioration. Figure 5.6 shows the state sequences, which correspond to the observation sequences, based on which the HMMs for the respective branches are trained. Section 4.2.3 describes how exactly those sequences are

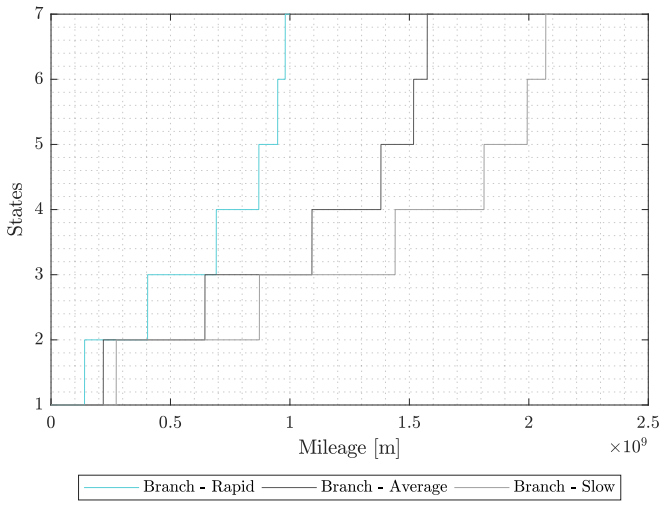


Figure 5.6.: Sequence of states for three branches.

derived in terms of their length and composition. The state composition[4] is

[4]Determins how many samples correspond the each HS

derived from a Poisson distribution, the length from a normal distribution. However, the selected models where chosen to represent average, lower and upper deterioration modes. All three of them contribute to the estimated value of the RUL, weighted by their likelihood, as described in section 4.3.3. Matlab offers a bunch of functions for standard HMMs, which where used to derive the branches. "hmmestimate" delivers the model with its system parameters $\lambda = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi})$. The mechanical deterioration is modeled with a straight left-right HMM, hence the states are not permitted to decrease.

Test Sequences of an MB-HMM

The testing sequences are derived in the exact same manner as the training data. Figure 5.7 shows three testing sequences that are used to assess the MB-HMM later on. Note that the illustrated sequences are actually not the

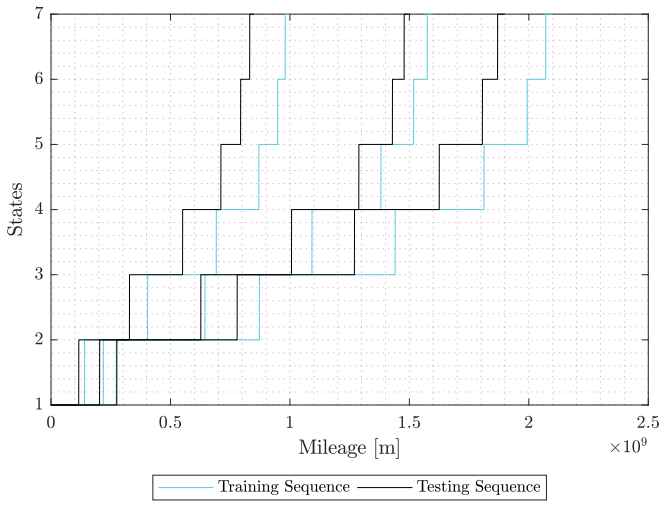


Figure 5.7.: Sequence of states for testing data and training data.

data, that is applied to the MB-HMM. They are only the corresponding state sequences, that are known because the data is generated synthetically. The

actual model uses the observations, represented by the posterior probability to estimate the state sequence. In section 5.3 these estimates are compared with the knowingly true state sequence, providing an out-of-sample error for the MB-HMM. Nonetheless, the shown state sequences illustrate the difference between the training and the testing data. They were selected in order to assess the MB-HMMs performance on differently degrading dampers. Therefore, three test sequences are derived, each of them representing one of the three branches, rapid, average and slow.

Assessment of Model Quality

The purpose the algorithm is to produce a statement about the RUL. Since the origin of the investigated data is a simulation, the complete data set is labeled. In other words, each produced sequence of observations has a corresponding sequence of states. In reality, a monitoring system provides only observation sequences, which are applied to the MB-HMM, whose objective is to determine the RUL. For the processing of simulated data, the desired output of the MB-HMM is known at any time, which enables an assessment of the output. The algorithm is applied with a sequence of observations which is iterative growing in length, where every iteration provides $\frac{1}{20}$ more of the entire sequence then the previous one. Figure 5.8 shows the iterative estimates of the three testing sequences from section 5.3. The turquoise line is the MB-HMMs output, which is the estimated value of the RUL according to equation 4.118, converted into a corresponding mileage value and evaluated at each iteration. The light grey lines are the confidence bounds, within which the true value should be with a likelihood of approximately 68%, i.e. the bound is determined by the standard derivation of the RUL according to equations 4.114 and 4.115. The black line illustrates the true RUL value, which is as already mentioned, known because the underlying testing data is synthetically generated. It is a linear curve, because the tested observation sequence is iterative increased in length, by a constant of one twentieth of the entire sequence. The small numbers, next to the data points that belong to each one of those iterations, indicate the most likely branch of the MB-HMM, which therefore has the biggest influence upon the respective RUL statement. The mapping of the numbers to the branches is as follows:

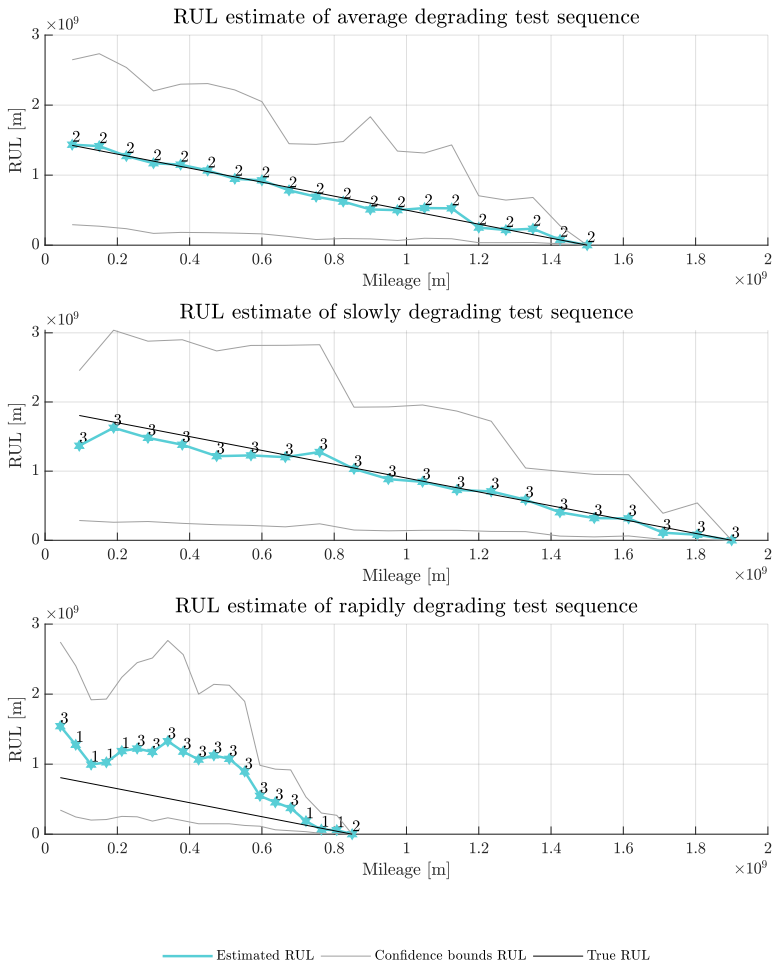


Figure 5.8.: RUL estimation for three different testing sequences, which represent either average, rapid or slow deterioration of the damper.

1. Rapid deterioration
2. Average deterioration
3. Slow deterioration

The first subplot shows the prognosis of an average degrading damper. Not surprisingly, the most likely branch is #2, because this HMM was trained with the average sequence. The models "best guess" is actually very precise, however the confidence bounds are rather big. The second subplot illustrates an estimation of a slowly degrading damper. Equal to the average degrading sequence, the most likely branch is always #3, which is the branch that was trained with the slowly degrading observation sequence. The MB-HMM has a rather hard time predicting the RUL of the rapidly degrading sequence, illustrated in the third subplot. Until about 80% of the total mileage is reached, it overestimates the true RUL significantly. Obviously the most likely branch changes several times throughout the lifetime. Only about one third of the iterations are calculated with the "right" model #1. Approaching the EoL, is hops back to the rapid deterioration branch, decreasing the models error. This kind of behavior is expected when different failure modes occur at the same time. However, the generated test sequence only represents one failure mode, i.e. the MB-HMM parameters have to be optimized in order to guarantee a better prediction of rapid deterioration. Although the model does not always deliver the right RUL, the true value lies within the confidence bounds at every iteration. It is basically a matter of model optimization, that can improve the result in terms of smaller confidence bound and a better estimate. Generally, the MB-HMM seems to be a promising method to predict the RUL of component, whose deterioration behavior is non-linear. Still, it is a data driven method, hence strongly dependent on the quality of data.

6. Conclusion and Prospect

In course of this thesis, an improved maintenance strategy is identified as one of the major benefits of digitization in railway industry. A sophisticated utilization of hardware and software, that comes with a so called "digital train", enables enormous potential for improvement. Besides other aspects, like increased safety or advanced possibilities within the development process, the condition based maintenance (CBM) strategy, builds upon smart concepts of big data handling. Provided by an on-board monitoring system of the digital train, a huge amount of data is collected, that needs proper treatment in order to make the whole system profitable. The processing of such data with the objective of condition prediction of mechanical components, is focus of this thesis.

A comparison of data driven and model based approaches leads to the selection of a machine learning method. Its advanced abilities in modeling non-linear and complex systems make them a perfect choice for the presented problem of health condition prediction. The discussion of fundamentals of machine learning shows the basic procedure, that leads to a successful implementation of such a statistical model. It includes the two substantial phases, learning and testing. The application condition prediction of mechanical components requires a method from the field of supervised learning.

Based on an extensive literature research as well as on expert counsel, hidden Markov models (HMMs) are chosen to be implemented. Since they are able to model stochastic processes resolved over time, a continuous deterioration of mechanical components, even of non-linear character, is represented very well. An HMM basically consists of a finite number of distinct states, which are usually unknown. Each constituent state emits one observation, out of a certain number of distinct observations. Based on the analysis of the observation, the state from which it is emitted can be determined in a probabilistic way. By analyzing not only one, but an entire chain of observations, which are emitted

within a certain time, the corresponding chain of states can be determined. Within that process both, the emission of a certain observation and the transition from one state to another, are modeled with probability distributions. The HMM represents a sequence of several two staged, stochastic events, which are linked with probabilities. For the use case condition prediction, the states represent the components health states (HSs) and the observations are certain features, that are requested to be most informative. An additional functionality is added to the HMM by training three models with diverse data sets. Each of them, representing either a rapid, average or slow deterioration mode of the analyzed component. The output of the so called multi-branch hidden Markov model (MB-HMM), is produced by means of Bayesian model averaging, allowing every single trained branch to contribute to the final result. Thereby the respective contributions are weighted by the likelihood of the branch. However, the basic version of each constituent HMM can only handle one-dimensional observations. A reasonable analysis of the components health condition is hardly possible with only one quantity. To tackle that problem, a classifier is used, that is supposed to obtain a one-dimensional measure that correlates well to the HS. Due to its outstanding ability of generalizing, a support vector machine (SVM) is chosen to be that classifier.

A SVM models mutual dependencies from certain statistical quantities and assigns the corresponding data set to a certain class. That way, a combination of several statistical quantities is converted to the desired one-dimensional observations. The SVM delivers a likelihood for each sample of the applied data set, of being produced by a faulty component. Hence, the SVMs output is a quantification of the components HS, which is then further processed by the MB-HMM.

Both of those, sub-sequentially performing statistical models are implemented, which is done by following through the above mentioned phases of machine learning. Therefore, the first step is to learn the models. Either one of them requires data from both, healthy and faulty components, without which supervised learning is certainly not feasible. This data is acquired by a multi-body simulation, which is executed with varying vehicle parameter sets. It is a linearized, three-dimensional state-space model of a railway vehicle, represented by its main components, namely the car body, two bogies and four wheel sets. It accounts for a linearized wheel rail contact, position and level of the track as well as curvature. The parameter variation is realized

through an iterative approach, where the parameter of the considered component is reduced step-by-step, simulating its deterioration. The produced sequences represent the entire lifetime of the component, assuming its HS evolution is Poisson distributed and the end of life (EoL) mileage is normal distributed. The parameter set for both of those distributions depend on the considered component, which is why from this point on, the procedure is specifically performed, only for the secondary vertical damper. Its parameter set is determined, based on expert knowledge and component tests. The basic approach for the condition prediction is similar for almost any component on the vehicle, however the data acquisition is not. Therefore, the simulation will only represent the degradation of one single component, which in this case is the secondary vertical damper. In conclusion, to successfully implement a condition prediction by basic MB-HMMs, a one-dimensional and labeled data set is required. This data set is synthetically generated by a multi-body simulation and thereafter classified by a SVM, whose output is a probability measure that indicates the components health.

The second phase is testing, where the learned models are applied with completely new, independent and unlabeled data, which is again generated by the multi-body simulation. In doing so, the SVM delivers a mapping of a three-dimensional feature data set to its corresponding health condition. After an appropriate pre-processing, which is first and foremost the application of a band-pass filter, a data set is being present, that consists of a sequence of HSs and another one of the observable likelihood of faultiness. The latter is applied to the MB-HMM, which detects the current health situation and predicts the most likely further propagation until the EoL. This procedure is repeatedly executed for every branch, increasing the length of the provided input data after each iteration. The output is a quantity that represents the time or mileage which is left, until the damper eventually brakes down. This quantity is also referred to as the remaining useful lifetime (RUL) and is produced by means of Bayesian averaging of all constituent branches. The applied data has a corresponding sequence of HSs, which is used for an assessment of the MB-HMM's performance.

The MB-HMM is tested with three different training sequences, with the objective of a selective assessment of the three deterioration branches rapid, average and slow. Therefore, each of these training sequences is chosen to be similar to one of the constituent branches. The results show, that the

performance of the MB-HMM is certainly acceptable, given that the RUL lies always between the conference bounds of the estimate. The model does a great job on average and slowly degrading dampers, however the rapid deterioration does not work out perfectly. At least, not until about 80% of the entire sequence is provided. This behavior is traced back to the nature of the algorithm that computes the most likely branch, which is the one that contributes most. Apparently, this selection does not work properly for the rapid deterioration. The algorithm compares the HMMs by means of their posterior probability, a measure, that quantifies the likelihood of an observation to be produced by the model. A practicable way of determining the best model, however it neither accounts the previously made decisions, nor has it any functionality that relates to a physical interpretation of e.g. a change in the deterioration branch. There are many possibilities of improving the MB-HMM, in order to optimize the estimation for all desired deterioration modes. A structured conditioning of the model parameters is a quick way, whereas an upgrade to a so called continuous HMM, or a hidden semi Markov model (HSMM) represents the extensive improvements. The continuous HMM is capable of processing continuously, rather than discrete distributed, real valued observations. Hence its not restricted to a finite set of observations, which actually is a more accurate representation of the reality, considering the inputs are provided by a monitoring system i.e. certainly not discrete. The term HSMM can frequently be found in literature and is referring to a different modeling of the state sojourn time, which enables a more precise adaption to the use case, that the HSMM is trained for. Yet another expansion of the basic HMM is the consideration of more than one emitted observation per state, which would make the classification by SVM obsolete. All of those proposed upgrades build upon the basic algorithms, presented in this thesis. However, they certainly require implementation of adjustments.

Although many assumptions have been made throughout the development process of the RUL algorithm, its performance on synthetic data provides very promising results. Testing the statistical models with real, rather than simulated data, is certainly way more challenging. Additional non-linear phenomena occur and make it even harder, to detect certain anomalies. However, the already largely acceptable MB-HMM, offers several improvements, which can optimize its output and even make it a considerable method for predicting real data.

Appendix

Appendix A.

The State Space Model

A.1. An Example of the Newton Euler Procedure

The procedure according to Newton Euler, which is applied to the model of a complete railway vehicle in section 4.1.3, is exemplary performed on a simplified vehicle, represented by a double oscillator, in order to provide details to the approach. Give a two mass system with two DoF according to A.1. For this simplified vehicle model, equations 4.3 and 4.4 yield [14]:

$$m_1\ddot{z}_1 + d_1\dot{z}_1 - d_1\dot{z}_2 + c_1z_1 - c_1z_2 = 0 \quad , \tag{A.1}$$

$$m_2\ddot{z}_2 + (d_1+d_2)\dot{z}_2 - d_1\dot{z}_1 + (c_1+c_2)z_2 - c_1z_1 = d_2\dot{u}_z + c_2u_z \quad , \tag{A.2}$$

which can easily be rearranged into matrix form, i.e.:

$$\begin{bmatrix} m_1 & 0 \\ 0 & m_2 \end{bmatrix} \begin{bmatrix} \ddot{z}_1 \\ \ddot{z}_2 \end{bmatrix} + \begin{bmatrix} d_1 & -d_1 \\ -d_1 & d_1+d_2 \end{bmatrix} \begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} + \begin{bmatrix} c_1 & -c_1 \\ -c_1 & c_1+c_2 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ c_2 & d_2 \end{bmatrix} \begin{bmatrix} u_z \\ \dot{u}_z \end{bmatrix} \quad ,$$

or in a more compact form as:

$$\boldsymbol{M}\ddot{\boldsymbol{z}} + \boldsymbol{D}\dot{\boldsymbol{z}} + \boldsymbol{C}\boldsymbol{z} = \boldsymbol{U}\boldsymbol{u} \quad , \tag{A.3}$$

with the mass matrix $\boldsymbol{M}$, damping matrix $\boldsymbol{D}$, stiffness matrix $\boldsymbol{C}$ and excitation matrix $\boldsymbol{B}$. As described in section 4.1.4, to solve the system of equations conveniently, a state-space transformation is necessary. The system of two

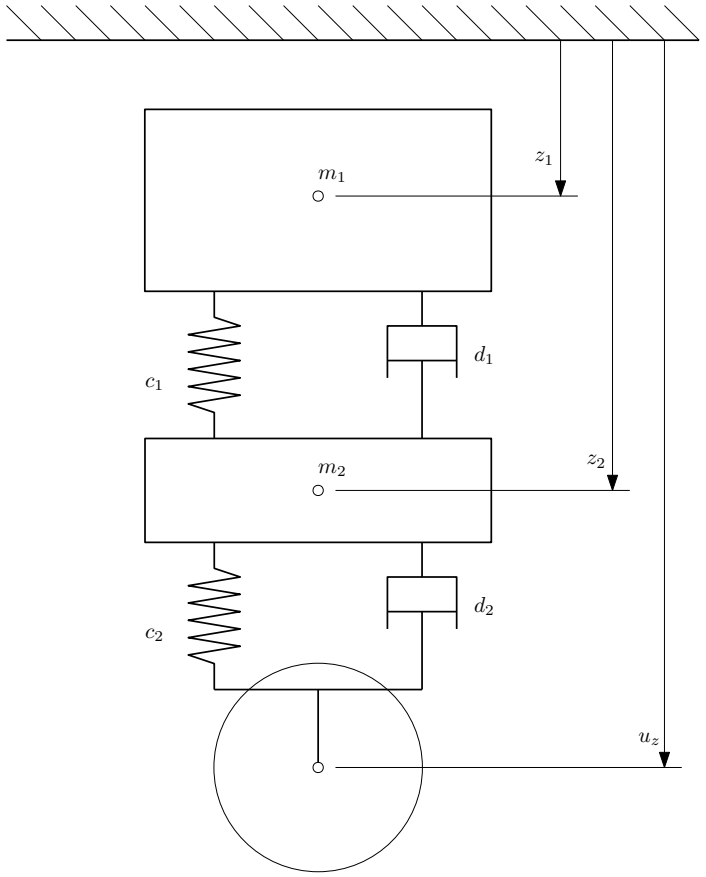


Figure A.1.: Simplified vehicle model, represented by a double oscillator with two DoF.

second order differential equations A.3, is converted into a system of four differential equations of first order, i.e. with

$$\boldsymbol{y} = \begin{bmatrix} z_1 \\ z_2 \\ \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} \quad , \quad \dot{\boldsymbol{y}} = \begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \\ \ddot{z}_1 \\ \ddot{z}_2 \end{bmatrix} \quad \text{and} \quad \bar{\boldsymbol{u}} = \begin{bmatrix} 0 \\ 0 \\ \dot{u}_z \\ u_z \end{bmatrix}$$

the new system can be written in the following form [14]:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & m_1 & 0 \\ 0 & 0 & 0 & m_2 \end{bmatrix} \begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \\ \ddot{z}_1 \\ \ddot{z}_2 \end{bmatrix} + \begin{bmatrix} 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \\ c_1 + c_2 & -c_2 & d_1 + d_2 & -d_2 \\ -c_2 & c_2 & -d_2 & d_2 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} = \\ = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & c_2 & d_2 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ u_z \\ \dot{u}_z \end{bmatrix},$$

or in a more compact form as:

$$\bar{\boldsymbol{M}}\dot{\boldsymbol{y}} + \bar{\boldsymbol{C}}\boldsymbol{y} = \bar{\boldsymbol{U}}\bar{\boldsymbol{u}} \quad . \tag{A.4}$$

The above equation rearranged yields the desired form, in order to solve the equation system numerically:

$$\begin{aligned} \dot{\boldsymbol{y}} &= -\bar{\boldsymbol{M}}^{-1}\bar{\boldsymbol{C}}\boldsymbol{y} + \bar{\boldsymbol{M}}^{-1}\bar{\boldsymbol{U}}\bar{\boldsymbol{u}} \\ \dot{\boldsymbol{y}} &= \boldsymbol{A}\boldsymbol{y} + \boldsymbol{B}\bar{\boldsymbol{u}} \quad . \end{aligned} \tag{A.5}$$

With $\boldsymbol{A}$ and $\boldsymbol{B}$ defined as:

$$\boldsymbol{A} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{c_1+c_2}{m_1} & \frac{c_2}{m_1} & -\frac{d_1+d_2}{m_1} & \frac{d_2}{m_1} \\ \frac{c_2}{m_2} & -\frac{c_2}{m_2} & \frac{d_2}{m_2} & -\frac{d_2}{m_2} \end{bmatrix} \quad \text{and} \quad \boldsymbol{B} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{c_2}{m_2} & \frac{d_2}{m_2} \end{bmatrix} \tag{A.6}$$

In the same manner, systems of any complexity can be solved. Several numerical solvers, implemented in various commercial software packages can handle this type of equation system.

A.2. Details to the Connection Equations

In chapter 4, connection forces and torques were introduced, which are responsible for the transmission of motion through the multi-body system. To

exemplary show an entry of the connection force or torque for one coordinate, they have to be dissolved. The following equations are the results of the vector equations 4.15 to 4.18

$$\boldsymbol{F}_{ckij} = \begin{bmatrix} F_{cxkij} \\ F_{cykij} \\ F_{czkij} \end{bmatrix} = \begin{bmatrix} c_{xkij} & 0 & 0 \\ 0 & c_{ykij} & 0 \\ 0 & 0 & c_{zkij} \end{bmatrix} \cdot \begin{bmatrix} x_{\hat{K}} - x_K + (\varphi_{\hat{K}} l_{z,\hat{K},kij} - \alpha_{\hat{K}} l_{y,\hat{K},kij}) - \\ y_{\hat{K}} - y_K + (\alpha_{\hat{K}} l_{x,\hat{K},kij} - \chi_{\hat{K}} l_{z,\hat{K},kij}) - \\ z_{\hat{K}} - z_K + (\chi_{\hat{K}} l_{y,\hat{K},kij} - \varphi_{\hat{K}} l_{x,\hat{K},kij}) - \\ - (\varphi_K l_{z,K,kij} - \alpha_K l_{y,K,kij}) \\ - (\alpha_K l_{z,K,kij} - \chi_K l_{z,K,kij}) \\ - (\chi_K l_{y,K,kij} - \varphi_K l_{x,K,kij}) \end{bmatrix} \tag{A.7}$$

$$\boldsymbol{F}_{dkij} = \begin{bmatrix} F_{dxkij} \\ F_{dykij} \\ F_{dzkij} \end{bmatrix} = \begin{bmatrix} d_{xI1} & 0 & 0 \\ 0 & d_{yI1} & 0 \\ 0 & 0 & d_{zI1} \end{bmatrix} \cdot \begin{bmatrix} \dot{x}_{\hat{K}} - \dot{x}_K + (\dot{\varphi}_{\hat{K}} l_{z,\hat{K},kij} - \dot{\alpha}_{\hat{K}} l_{y,\hat{K},kij}) - \\ \dot{y}_{\hat{K}} - \dot{y}_K + (\dot{\alpha}_{\hat{K}} l_{x,\hat{K},kij} - \dot{\chi}_{\hat{K}} l_{z,\hat{K},kij}) - \\ \dot{z}_{\hat{K}} - \dot{z}_K + (\dot{\chi}_{\hat{K}} l_{y,\hat{K},kij} - \dot{\varphi}_{\hat{K}} l_{x,\hat{K},kij}) - \\ - (\dot{\varphi}_K l_{z,K,kij} - \dot{\alpha}_K l_{y,K,kij}) \\ - (\dot{\alpha}_K l_{z,K,kij} - \dot{\chi}_K l_{z,K,kij}) \\ - (\dot{\chi}_K l_{y,K,kij} - \dot{\varphi}_K l_{x,K,kij}) \end{bmatrix} \tag{A.8}$$

$$
\begin{aligned}
\boldsymbol{M}_{ckij} = \begin{bmatrix} M_{cxkij} \\ M_{cykij} \\ M_{czkij} \end{bmatrix} = \begin{bmatrix} l_{x,K,kij} \\ l_{y,K,kij} \\ l_{z,K,kij} \end{bmatrix} \times \begin{bmatrix} F_{cxkij} \\ F_{cykij} \\ F_{czkij} \end{bmatrix} = \begin{bmatrix} l_{y,K,kij} c_{zkij} \big(z_{\hat{K}} - z_K + \\ l_{z,K,kij} c_{xkij} \big(x_{\hat{K}} - x_K + \\ l_{x,K,kij} c_{ykij} \big(y_{\hat{K}} - y_K + \end{bmatrix} \\
+ (\chi_{\hat{K}} l_{y,\hat{K},kij} - \varphi_{\hat{K}} l_{x,\hat{K},kij}) - (\chi_K l_{y,K,kij} - \varphi_K l_{x,K,kij})) - \\
+ (\varphi_{\hat{K}} l_{z,\hat{K},kij} - \alpha_{\hat{K}} l_{y,\hat{K},kij}) - (\varphi_K l_{z,K,kij} - \alpha_K l_{y,K,kij})) - \\
+ (\alpha_{\hat{K}} l_{x,\hat{K},kij} - \chi_{\hat{K}} l_{z,\hat{K},kij}) - (\alpha_K l_{z,K,kij} - \chi_K l_{z,K,kij})) - \\
- l_{z,K,kij} c_{ykij} \big(y_{\hat{K}} - y_K + (\alpha_{\hat{K}} l_{x,\hat{K},kij} - \chi_{\hat{K}} l_{z,\hat{K},kij}) - \\
- l_{x,K,kij} c_{zkij} \big(z_{\hat{K}} - z_K + (\chi_{\hat{K}} l_{y,\hat{K},kij} - \varphi_{\hat{K}} l_{x,\hat{K},kij}) - \\
- l_{y,K,kij} c_{xkij} \big(x_{\hat{K}} - x_K + (\varphi_{\hat{K}} l_{z,\hat{K},kij} - \alpha_{\hat{K}} l_{y,\hat{K},kij}) - \\
- (\alpha_K l_{z,K,kij} - \chi_K l_{z,K,kij})) \\
- (\chi_K l_{y,K,kij} - \varphi_K l_{x,K,kij})) \\
- (\varphi_K l_{z,K,kij} - \alpha_K l_{y,K,kij})) \big]
\end{aligned}
\tag{A.9}
$$

$$
\begin{aligned}
\boldsymbol{M}_{dkij} = \begin{bmatrix} M_{dxkij} \\ M_{dykij} \\ M_{dzkij} \end{bmatrix} = \begin{bmatrix} l_{x,K,kij} \\ l_{y,K,kij} \\ l_{z,K,kij} \end{bmatrix} \times \begin{bmatrix} F_{dxkij} \\ F_{dykij} \\ F_{dzkij} \end{bmatrix} = \begin{bmatrix} l_{y,K,kij} d_{zkij} \big(\dot{z}_{\hat{K}} - \dot{z}_K + \\ l_{z,K,kij} d_{xkij} \big(\dot{x}_{\hat{K}} - \dot{x}_K + \\ l_{x,K,kij} d_{ykij} \big(\dot{y}_{\hat{K}} - \dot{y}_K + \end{bmatrix} \\
+ (\dot{\chi}_{\hat{K}} l_{y,\hat{K},kij} - \dot{\varphi}_{\hat{K}} l_{x,\hat{K},kij}) - (\dot{\chi}_K l_{y,K,kij} - \dot{\varphi}_K l_{x,K,kij})) - \\
+ (\dot{\varphi}_{\hat{K}} l_{z,\hat{K},kij} - \dot{\alpha}_{\hat{K}} l_{y,\hat{K},kij}) - (\dot{\varphi}_K l_{z,K,kij} - \dot{\alpha_K} l_{y,K,kij})) - \\
+ (\dot{\alpha}_{\hat{K}} l_{x,\hat{K},kij} - \dot{\chi}_{\hat{K}} l_{z,\hat{K},kij}) - (\dot{\alpha}_K l_{z,K,kij} - \dot{\chi}_K l_{z,K,kij})) - \\
- l_{z,K,kij} d_{ykij} \big(\dot{y}_{\hat{K}} - \dot{y}_K + (\dot{\alpha}_{\hat{K}} l_{x,\hat{K},kij} - \dot{\chi}_{\hat{K}} l_{z,\hat{K},kij}) - \\
- l_{x,K,kij} d_{zkij} \big(\dot{z}_{\hat{K}} - \dot{z}_K + (\dot{\chi}_{\hat{K}} l_{y,\hat{K},kij} - \dot{\varphi}_{\hat{K}} l_{x,\hat{K},kij}) - \\
- l_{y,K,kij} d_{xkij} \big(\dot{x}_{\hat{K}} - \dot{x}_K + (\dot{\varphi}_{\hat{K}} l_{z,\hat{K},kij} - \dot{\alpha}_{\hat{K}} l_{y,\hat{K},kij}) - \\
- (\dot{\alpha}_K l_{z,K,kij} - \dot{\chi}_K l_{z,K,kij})) \\
- (\dot{\chi}_K l_{y,K,kij} - \dot{\varphi}_K l_{x,K,kij})) \\
- (\dot{\varphi}_K l_{z,K,kij} - \dot{\alpha}_K l_{y,K,kij})) \big]
\end{aligned}
\tag{A.10}
$$

Appendix B.

Pseudo Code

This section provides a conclusion of the pseudo code for the framework presented in section 4.3.3. It consists of the following HMM related algorithms

- Forward algorithm
- Viterbi algorithm
- EM algorithm (Baum-Welch algorithm)
- Bayesian model averaging
- Complete RUL procedure

Algorithm 6 Solving the evaluation problem with the forward algorithm

function FORWARD-ALGORITHM(o, λ)
    create a forward probability matrix α
    for state i from 1 to N **do**
        $\alpha[i,1] \leftarrow \pi_i \cdot b_i(o_1)$ ▷ Initialization
    for time t from 1 to $T-1$ **do**
        for state j from 1 to N **do**
            $\alpha[j,t+1] \leftarrow \sum_{i=1}^{N} \alpha[i,t] \cdot a_{ij} \cdot b_j(o_{t+1})$ ▷ Recursion
    $\alpha[N,T] \leftarrow \sum_{i=1}^{N} \alpha[i,T] \cdot a_{iN}$ ▷ Termination
    return $\alpha[N,T]$

Algorithm 7 Solving the decoding problem with the Viterbi algorithm

function VITERBI-ALGORITHM($\boldsymbol{o}$, λ)
    create a path probability matrix δ
    create backtracking matrix ψ
    create most likely state path q^*
    for state i from 1 to N **do**
        $\delta[i, 1] \leftarrow \pi_i \cdot b_i(o_1)$ ▷ Initialization
        $\psi[i, 1] \leftarrow 0$
    for time t from 1 to $T - 1$ **do**
        for state j from 1 to N **do**
            $\delta[j, t + 1] \leftarrow \max_{i=1}^{N} \delta[i, t] \cdot a_{ij} \cdot b_j(o_{t+1})]$ ▷ Recursion
            $\psi[j, t + 1] \leftarrow \text{argmax}_{i=1}^{N} \delta[i, t] \cdot a_{ij}$
    $\delta[N, T] \leftarrow \max_{i=1}^{N} \delta[i, T] \cdot a_{iN}$ ▷ Termination
    $\psi[N, T] \leftarrow argmax_{i=1}^{N} \delta[i, T] \cdot a_{iN}$
    for time t from $T - 1$ to 1 **do**
        $q^*[1, t] \leftarrow \psi[q^*[t + 1], t + 1]$ ▷ Backtracking
    return q^*

Algorithm 8 Solving the learning problem with the EM algorithm

function EM-ALGORITHM($\boldsymbol{o}$, a_{est}, b_{est}, π_{est}, $Iter$)
  for iterations $iter$ from 1 to $Iter$ **do**
    create probability arrays α, β, γ, ζ
    if $iter \leq 1$ **then**
      $a \leftarrow a_{est} \quad b \leftarrow b_{est} \quad \pi \leftarrow \pi_{est}$
    else
      $a \leftarrow \hat{a} \quad b \leftarrow \hat{b} \quad \pi \leftarrow \hat{\pi}$
    for state i from 1 to N **do**
      $\alpha[i,1] \leftarrow \pi_i \cdot b_i(o_1)$
      $\beta[i,1] \leftarrow 1$ ▷ Initialization
    for time t from 1 to $T-1$ **do**
      for state j from 1 to N **do**
        $\alpha[j,t+1] \leftarrow \sum_{i=1}^{N} \alpha[i,t] \cdot a_{ij} \cdot b_j(o_{t+1})]$ ▷ Forward recursion
    for time t from $T-1$ to 1 **do**
      for state i from 1 to N **do**
        $\beta[i,t] \leftarrow \sum_{j=1}^{N} a_{ij} \cdot b_j(o_t) \cdot \beta[j,t+1]$ ▷ Backward recursion
    for time t from 1 to T **do**
      for state j from 1 to N **do**
        $\gamma[j,t] \leftarrow \frac{\alpha[j,t] \cdot \beta[j,t]}{\sum_{i=1}^{N} \alpha[i,t] \cdot \beta[i,t]}$
    for time t from 1 to $T-1$ **do**
      for state i from 1 to N **do**
        for state j from 1 to N **do**
          $\zeta[i,j,t] \leftarrow \frac{\alpha[i,t] \cdot a_{ij} \cdot b_j(o_{t+1})] \cdot \beta[j,t]}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha[i,t] \cdot a_{ij} \cdot b_j(o_{t+1})] \cdot \beta[j,t]}$
    for state i from 1 to N **do**
      $\hat{\pi}_i \leftarrow \gamma[i,1]$
      for state j from 1 to N **do**
        $\hat{a}_{i,j} \leftarrow \frac{\sum_{t=1}^{T-1} \zeta[i,j,t]}{\sum_{t=1}^{T-1} \gamma[i,t]}$
    for state j from 1 to N **do**
      for observation k from 1 to K **do**
        $validObs \leftarrow (\boldsymbol{o} == V_k)$
        $\hat{b}_j(V_k) \leftarrow \frac{\sum_{t=1}^{T} (\gamma[j,t] \circ validObs)}{\sum_{t=1}^{T} \gamma[j,t]}$
  return $\lambda \leftarrow (\hat{\pi}, \hat{a}, \hat{b})$

Algorithm 9 MB-HMM contribution with Bayesian model averaging

function BAYESIANMODELAVERAGING($P(\boldsymbol{o}_{test}|\lambda)$, $P(\lambda)$,RUL_{ψ}^{l})
  for branch l from 1 to L **do**
    $P(\lambda_l|\boldsymbol{o}_{test}) \leftarrow \frac{P(\boldsymbol{o}_{test}|\lambda_l)\cdot P(\lambda_l)}{\sum_{l=1}^{L} P(\boldsymbol{o}_{test}|\lambda_l)\cdot P(\lambda_l)}$
    $RUL_{\psi} \leftarrow \sum_{l=1}^{L} RUL_{\psi}^{l} \cdot P(\lambda|\boldsymbol{o}_{test})$
  return RUL

Algorithm 10 RUL

procedure RUL-ESTIMATE($\boldsymbol{o}_{train}$,$\boldsymbol{o}_{test}$,a_{est},b_{est},π_{est},$Iter$)
  for branch l from 1 to L **do**
    function EM-ALGORITHM($\boldsymbol{o}^{l}_{train}$,$a_{est}$,$b_{est}$,$\pi_{est}$,$Iter$)
      return $\lambda_l \leftarrow (\hat{\pi},\hat{a},\hat{b})$
    function FORWARD-ALGORITHM($\boldsymbol{o}_{test}$, λ_l)
      return $P(\boldsymbol{o}_{test}|\lambda_l)$
    function VITERBI-ALGORITHM($\boldsymbol{o}_{test}$, λ_l)
      return $\boldsymbol{q}^*$
    $currentState \leftarrow q^*[end]$
    $numCurrentState \leftarrow length(find(\boldsymbol{q}^* == currentState))$
    $numPreviousStates \leftarrow length(\boldsymbol{q}^*)$
    function HMMGENERATE(λ_l)
      return $\boldsymbol{o}_{gen}$]
    function VITERBI-ALGORITHM($\boldsymbol{o}_{gen}$, λ_l)
      return $\boldsymbol{q}^*$
    create time step matrix Δt
    create RUL matrices $\boldsymbol{RUL}_{\mu}$,$\boldsymbol{RUL}_{upper}$,$\boldsymbol{RUL}_{lower}$
    for gen y from 1 to Y **do**
      for state $ifind$ from 1 to N **do**
        $\Delta t[y, ifind] \leftarrow find(\boldsymbol{q}^*[y] == ifind,$ last sample$)$
    $\Delta t \leftarrow \Delta t[:, currentState : N-1]$
    $\Delta t[:, currentState] - numPreviousStates$
    for j from 1 to N **do**
      $\boldsymbol{RUL}^{l}_{\mu}[j] \leftarrow mean(\Delta t[:, j])$
      $\boldsymbol{RUL}^{l}_{upper}[j] \leftarrow mean(\Delta t[:, j] + std(\Delta t[:, j])$
      $\boldsymbol{RUL}^{l}_{lower}[j] \leftarrow mean(\Delta t[:, j] - std(\Delta t[:, j])$
  function BAYESIANMODELAVERAGING($P(\boldsymbol{o}_{test}|\lambda_l)$,$P(\lambda_l)$,$\boldsymbol{RUL}^{l}_{\mu}$,$\boldsymbol{RUL}^{l}_{upper}$,$\boldsymbol{RUL}^{l}_{lower}$)
    return $\boldsymbol{RUL}_{\mu}$,$\boldsymbol{RUL}_{upper}$,$\boldsymbol{RUL}_{lower}$
  $RUL_{\mu} \leftarrow \sum_{j=1}^{N} RUL_{\mu}[j]$
  $RUL_{upper} \leftarrow \sum_{j=1}^{N} RUL_{upper}[j]$
  $RUL_{lower} \leftarrow \sum_{j=1}^{N} RUL_{lower}[j]$
  return RUL_{μ}, RUL_{upper}, RUL_{lower}

Bibliography

[1] Yaser S. Abu-Mostafa, M. Magdon-Ismail, and H.T.n Lin. *Learning from Data: A Short Course*. AMLbook.com, 2012. ISBN: 9781600490064 (cit. on pp. 15–17, 54).

[2] Heinrich Dubbel. *DUBBEL: Taschenbuch für den Maschinenbau*. Springer-Verlag, 2013 (cit. on pp. 32, 33, 35, 38, 42).

[3] S. D. Essinger and G. L. Rosen. "An introduction to machine learning for students in secondary education." In: *2011 Digital Signal Processing and Signal Processing Education Meeting (DSP/SPE)*. Jan. 2011, pp. 243–248. DOI: 10.1109/DSP-SPE.2011.5739219 (cit. on p. 14).

[4] Micaela Caserza Magro et.al. "CBM for a fleet of railway vehicles: infrastructure and algorithms." In: *ACTA IMEKO*. Vol. 5. 4. Dec. 2016, pp. 56–63. DOI: http://dx.doi.org/10.21014/acta_imeko.v5i4.412 (cit. on p. 2).

[5] Sang Chan Park et.al. "Application of IoT for the Maintaining Rolling Stocks." In: *Quality Innovation Prosperity*. Vol. 21. 2. Oct. 2017. DOI: 10.12776/QIP.V21I2.887 (cit. on p. 2).

[6] Gernot A Fink. *Markov models for pattern recognition: from theory to applications*. Springer Science & Business Media, 2014 (cit. on pp. 61, 65, 66, 69, 70, 72, 73).

[7] Luca Dimiccoli Francesco Cartella Jan Lemeire and Hichem Sahli. "Hidden Semi-Markov Models for Predictive Maintenance." In: *Mathematical Problems in Engineering* 2015 (2015), p. 23. URL: http://dx.doi.org/10.1155/2015/278120 (cit. on p. 24).

[8] Bernhard Girstmair, Andreas Haigermoser, and Justinian Rosca. "Combination of Data-driven Feature Selection Methods with Domain Knowledge for Diagnosis of Railway Vehicles." In: *Annual Conference of the Prognostics and Health Management Society 2017*. Vol. 8. 015. Aug. 2017, p. 10 (cit. on p. 21).

[9] Dr. Andreas Haigermoser. *Skriptum zur Vorlesung: Schienenfahrzeuge*. Graz University of Technology, 2002 (cit. on pp. 7, 25, 29, 36–38, 40, 41, 89).

[10] Jennifer A Hoeting et al. "Bayesian model averaging." In: *Proceedings of the AAAI Workshop on Integrating Multiple Learned Models*. Vol. 335. Citeseer. 1998, pp. 77–83 (cit. on p. 79).

[11] Joachim Ihme. *Schienenfahrzeugtechnik*. Springer, 2016 (cit. on p. 36).

[12] H. K. Jun and J. H. Kim. "Life cycle cost modeling for railway vehicle." In: *2007 International Conference on Electrical Machines and Systems (ICEMS)*. Oct. 2007, pp. 1989–1994 (cit. on p. 10).

[13] Joost Jacques Kalker. "On the rolling contact of two elastic bodies in the presence of dry friction." PhD thesis. TU Delft, Delft University of Technology, 1967 (cit. on pp. 37, 38).

[14] Prof. Dr.-Ing. habil. Katrin Ellermann. *Mehrkörperdynamik: Skriptum zur Vorlesung*. Graz University of Technology, 2015 (cit. on pp. 48, 106, 108).

[15] Rudolf Kruse et al. *Computational Intelligence-Eine methodische Einführung in Künstliche Neuronale Netze, Evolutionäre Algorithmen, Fuzzy-Systeme und Bayes-Netze. 1*. 2011 (cit. on p. 64).

[16] Thanh Trung Le, Florent Chatelain, and Christophe Bérenguer. "Multi-branch hidden Markov models for remaining useful life estimation of systems under multiple deterioration modes." In: *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability* 230.5 (Jan. 2016), pp. 473–484. DOI: 10.1177/1748006X15624584. URL: https://hal.archives-ouvertes.fr/hal-01260906 (cit. on pp. 24, 78, 79).

[17] K. Medjaher, D. A. Tobon-Mejia, and N. Zerhouni. "Remaining Useful Life Estimation of Critical Components With Application to Bearings." In: *IEEE Transactions on Reliability* 61.2 (June 2012), pp. 292–302. ISSN: 0018-9529. DOI: 10.1109/TR.2012.2194175 (cit. on p. 24).

[18] H Mikat, AM Siddiolo, and M Buderath. "Virtual Framework for Validation and Verification of System Design Requirements to enable Condition Based Maintenance." In: *First European Conference of the Prognostics and Health Management Society*. 2012 (cit. on pp. 3, 4).

[19] Javier R Movellan. "Tutorial on hidden Markov models." In: *Machine perception laboratory online tutorials* (2003) (cit. on pp. 24, 61, 65, 76).

[20] L. R. Rabiner. "A tutorial on hidden Markov models and selected applications in speech recognition." In: *Proceedings of the IEEE* 77.2 (Feb. 1989), pp. 257–286. ISSN: 0018-9219. DOI: 10.1109/5.18626 (cit. on pp. 24, 62, 63, 65–67, 69–72, 74–76, 82).

[21] Bernhard Schölkopf et al. "Support vector method for novelty detection." In: *Advances in neural information processing systems*. 2000, pp. 582–588 (cit. on pp. 51, 54).

[22] Aelita Skarzauskiene and Marius Kalinauskas. *The Future Potential of Internet of Things*. Social Technologies, Vol 2, No 1. 2012. URL: https://www3.mruni.eu/ojs/social-technologies/article/view/148/141 (cit. on p. 2).

[23] David R Westhead, MS Vijayabaskar, and Westhead. *Hidden Markov Models*. Springer, 2017 (cit. on p. 70).

[24] Achmad Widodo and Bo-Suk Yang. "Support vector machine in machine condition monitoring and fault diagnosis." In: *Mechanical systems and signal processing* 21.6 (2007), pp. 2560–2574 (cit. on pp. 50, 52, 54).