Predictive Health Monitoring for Aircraft Systems using Decision Trees

Mike Gerdes

Licentiate Thesis
Division of Fluid and Mechatronic Systems
Department of Management and Engineering
Linköping University
Predictive Health Monitoring for Aircraft Systems using Decision Trees

Mike Gerdes
Abstract

Unscheduled aircraft maintenance causes a lot problems and costs for aircraft operators. This is due to the fact that aircraft cause significant costs if flights have to be delayed or canceled and because spares are not always available at any place and sometimes have to be shipped across the world. Reducing the number of unscheduled maintenance is thus a great costs factor for aircraft operators. This thesis describes three methods for aircraft health monitoring and prediction; one method for system monitoring, one method for forecasting of time series and one method that combines the two other methods for one complete monitoring and prediction process. Together the three methods allow the forecasting of possible failures. The two base methods use decision trees for decision making in the processes and genetic optimization to improve the performance of the decision trees and to reduce the need for human interaction. Decision trees have the advantage that the generated code can be fast and easily processed, they can be altered by human experts without much work and they are readable by humans. The human readability and modification of the results is especially important to include special knowledge and to remove errors, which the automated code generation produced.

Keywords: Condition Monitoring, Condition Prediction, Failure Prediction, Decision Trees, Genetic Algorithm, Fuzzy Decision Tree Evaluation, System Monitoring, Aircraft Health Monitoring
Acknowledgment

I thank all persons who made this thesis possible. Professor Scholz and Bernhard Randerath for setting up the PAHMIR project (Preventive Aircraft Health Monitoring for Integrated Reconfiguration) and supporting me during all the time during my work. I also thank professor Petter Krus for making it possible to be a PhD. student at the Linköpings Universitet and for supporting me on my way. Finally I would like to thank Philotech GmbH that they made it possible for me to leave my job for a few years and return after my work at the HAW Hamburg.

The research for this thesis is sponsored by the government of Hamburg, Ministry for Economics and Labor (Behörde für Wirtschaft und Arbeit - BWA) as part of the Aviation Research Programme Hamburg (LuFo Hamburg).
Foreword

This thesis presents a concept for adaptable predictive aircraft health monitoring with decision trees. The project (PAHMIR - Preventive Aircraft Health Monitoring with Integrated Reconfiguration) that lead to this dissertation was started in 2008 as a cooperative project between Hamburg University of Applied sciences and Airbus Operations GmbH.

The organization of the dissertation is as followed:

Introduction The introduction chapter shall give the reader an understanding of the problem, the motivation for the research, the research background and the solution concept. In the begin the section describes the project in which the research was done and which gave the motivation for the research. This is followed by an explanation of the motivation and why research work was necessary. This is enhanced by a full description of the objectives of the research. The section closes with a review of the concepts that were applied to solve the problem and reach the objectives.

Theoretical Background The second section explains the theoretical background of the different concepts which are used for the concept. Those are: decision trees, heuristic optimization, signal analysis, condition monitoring and time series analysis. The order of the topics is based on the order how they are later used in the concept. The section closes with a summary.

Condition Monitoring The condition monitoring section contains the first part of the developed concept to solve the initial problems. It explains the process and how the methods that were presented in the previous chapter were applied to solve a part of the problem.

Condition Prediction Condition prediction is the second part of the concept and is explained in the fourth section. The section is in the same way structured like the previous section. The process of condition
prediction is explained and it is shown how the different methods work together to get a prediction of the system condition.

**Failure Prediction** Failure prediction is the combination of condition monitoring and condition prediction to forecast when a failure will happen. This section describes how the two previous processes can be combined to provide a complete process for failure prediction.

**Experiments** The experiments section shows how feasible and usable the developed concepts really are. The section is divided up into the evaluation of the condition monitoring concepts and the evaluation of the condition prediction. The two concepts use different experimental setups for the evaluation.

**Conclusions** The conclusions section summarizes the results and shows where future work can still be done.
This subsection gives an overview about the papers that were published during the writing of this thesis and to which the thesis refers during various points in the text. The papers are ordered in the chronological order in which they were published.

Reducing Delays Caused by Unscheduled Maintenance and Cabin Reconfiguration The first published paper analyses causes and costs for aircraft delays due to faults in the air conditioning system. The analysis includes an analysis of the duration of delays and how the duration and thus the costs can be reduced by using preventive maintenance. Additionally the paper discusses different method to reduce the duration for cabin reconfiguration. It was published during the International Workshop on Aircraft System Technologies 2009 (AST2009) in Hamburg[1].

Feature Extraction and Sensor Optimization for Condition Monitoring of Recirculation Fans and Filters The second published paper concentrates on feature extraction and sensor optimization using decision trees. A decision tree is used for sorting the features based on information gain. Then the sensors that produce the important feature can be made redundant, while other can be neglected. The second paper was published in 2010 during the Deutschen Luft- und Raumfahrt Kongress (DGLR2009) in Aachen[2].

Parameter Optimization for Automated Signal Analysis for Condition Monitoring of Aircraft Systems Following the feature extraction and sensor optimization is a concept for automatically selecting signal analysis methods and parameters to generate features for the decision tree calculation. A number of different signal analysis methods are selected and then an optimization algorithm is used to select the best signal analysis methods to generate feature for a given decision problem.
The paper compares different optimization algorithms and compares the speed and accuracy of those using data from an Airbus test rig for air conditioning. This paper was published during the International Workshop on Aircraft System Technologies 2011 (AST2011) in Hamburg[3].

**Fuzzy Condition Monitoring of Recirculation Fans and Filters** Decision trees normally yield only hard discreet results. In this paper a method was published, which uses a normal decision tree and weights decisions taken to create a fuzzy result. The result gives the user a feedback on how likely other results are and what happens to the results if a parameter slightly changes. The paper was published in the CEAS Aeronautical Journal in 2011 and presented on the Deutschen Luft- und Raumfahrt Kongress (DGLR2011) in Bremen[4].

**Decision Trees and Genetic Algorithms for Condition Monitoring Forecasting of Aircraft Air Conditioning** The last published paper looks at the use of decision trees to predict future values in a time series. A decision tree is trained on past data and then uses features of a time series to decide, which extrapolation method for the next data points is the best for the current time series. The paper also shows some theoretical experiments and shows how the method can be improved for different problems. The last paper was published 2013 in the Expert Systems With Applications journal[5].


Contents

1 Introduction 1
  1.1 Preventive Aircraft Health Monitoring for Integrated Re-
    configuration (PAHMIR) .............................. 1
  1.2 Motivation ............................................. 2
  1.3 Research Goal ......................................... 3
  1.4 Concept ................................................ 4
    1.4.1 Condition Monitoring ............................. 5
    1.4.2 Condition Prediction ............................... 7
    1.4.3 Failure Prediction ................................. 9

2 Theoretical Background 11
  2.1 Condition Monitoring ................................. 11
  2.2 Signal Analysis ......................................... 17
  2.3 Trend Series Analysis & Forecasting .................. 20
    2.3.1 Simple Linear Regression ........................ 23
    2.3.2 Multiple Regression ................................ 24
    2.3.3 Moving Average Model ............................. 25
    2.3.4 Exponential Smoothing ............................ 26
    2.3.5 Box-Jenkins ....................................... 26
    2.3.6 Other methods ..................................... 29
  2.4 Decision Trees ........................................... 29
  2.5 Local search and optimization ........................ 32
  2.6 Literature Review ....................................... 34
    2.6.1 Look-Ahead Based Fuzzy Decision Tree Induction 34
    2.6.2 Fuzzy Decision Tree for Data Mining of Time
        Series Stock Market Databases ....................... 34
    2.6.3 Induction of Decision Trees ....................... 34
    2.6.4 A complete fuzzy decision tree technique ........ 35
2.6.5 Intelligent stock trading system by turning point confirming and probabilistic reasoning .... 35
2.6.6 Continuous Trend-Based Classification of Streaming Time Series ......................... 35
2.6.7 A Neural Stock Price Predictor using Quantitative Data .................................. 35
2.6.8 A Neural Network Approach to Condition Based Maintenance: Case Study of Airport Ground Transportation Vehicles ......................... 35
2.6.9 A Multi-Agent Fault Detection System for Wind Turbine Defect Recognition and Diagnosis .... 36
2.6.10 Sensor fusion of a railway bridge load test using neural networks ....................... 36
2.6.11 Automated Rule Extraction for Engine Vibration Analysis ............................... 36
2.6.12 A survey of outlier detection methodologies ............................................... 36
2.6.13 Using Genetic Algorithms for Adapting Approximation Functions .................. 37
2.6.14 Decision tree classifier for network intrusion detection with GA-based feature selection .... 37

3 Condition Monitoring 39
3.1 Data Requirements ........................................ 41
3.2 Training Process ......................................... 42
3.2.1 Data samples ............................................ 42
3.2.2 Data Labeling .......................................... 43
3.2.3 Preprocessing ............................................ 44
3.2.4 Calculate Decision Tree ............................... 49
3.2.5 Optimize Decision Tree .............................. 50
3.3 Monitoring Process ....................................... 50
3.4 Summary .................................................. 50

4 Condition Prediction 53
4.1 Training Process ......................................... 55
4.1.1 Data Samples ............................................ 55
4.1.2 Data Classification ..................................... 57
4.1.3 Data Preprocessing ................................. 58
4.1.4 Decision Tree Calculation ......................... 59
4.1.5 Optimization .......................................... 60
V Decision Trees and Genetic Algorithms for Condition Monitoring Forecasting of Aircraft Air Conditioning 157
1

Introduction

This section gives the reader an overview about the motivation and concept discussed in this document.

1.1 Preventive Aircraft Health Monitoring for Integrated Reconfiguration (PAHMIR)

The PAHMIR (Preventive Aircraft Health Monitoring for Integrated Reconfiguration) project was the research environment and base for this work. PAHMIR was a cooperative research project between Airbus Operations GmbH and Hamburg University of Applied Sciences (HAW Hamburg). The project was funded by the city of Hamburg and had a duration of 3.5 years. Start was January 2008 and end was June 2011. All research work was done during this period. Goal of PAHMIR was to analyze existing in-service aircraft maintenance data, develop a predictive aircraft health monitoring system and analyze how such a system might be integrated into a dynamic cabin concept. Concepts for condition monitoring, condition prediction and in door localization were developed and tested in experiments in the project.

The goals of PAHMIR are:

- Reduction of unscheduled maintenance
- Advanced failure prediction
- Condition monitoring
- Ability to better plan maintenance
Predictive Health Monitoring for Aircraft Systems using Decision Trees

• Improve cabin reconfiguration

1.2 Motivation

One goal of PAHMIR was to prevent and forecast failures. The main drivers for the development of a failure prediction concept are the costs of a delay of an aircraft departure or arrival. Delays can be caused by unscheduled maintenance between aircraft arrival and aircraft departure. Failure prediction shall allow the aircraft operator to repair or replace a system during scheduled maintenance, if the system is not yet broken but will be before the next scheduled maintenance. Figure 1.1 shows the handling of an aircraft fault without predictive health monitoring (failure prediction).

![Figure 1.1](image)

**Figure 1.1** Unscheduled maintenance without failure forecasting

The maintenance case in Figure 1.1 is like this: a fault happens in flight. Sensors detect the fault and report the fault to the cockpit. The pilot/aircraft sends a maintenance request to the airport. A maintenance mechanic checks the aircraft, when it is on ground. The mechanic performs a fault search and a fault diagnosis. Spare parts are ordered and a repair plan is made after the fault has been identified. When the spare parts arrive, it is possible to do the repair. The aircraft is ready again after the repair. It is possible that the fault identification, diagnostics and spare parts management take too much time, so that the aircraft departure is delayed or even canceled. A delay causes significant costs for an aircraft operator.

In the ideal case (to which PAHMIR is a step) most faults that will
occur are repaired during scheduled maintenance (Figure 1.2). It is still possible that a fault occurs. Sensors and fault detection systems identify and diagnose the fault, if a fault occurs during flight. The aircraft sends a fault report and diagnosis to the ground, where a maintenance mechanic gets the spare parts and prepares the repair. The fault is repair after the aircraft lands. Delays can be prevented by repairing future faults in the hanger. This reduces the number of unscheduled maintenance cases.

Figure 1.2 Unscheduled maintenance with failure forecasting

The costs of a delay can be quite large if the delay is long or the flight is canceled. [1] shows an analysis of the costs of a delay and what can be saved by forecasting faults and do repairs during scheduled maintenance.

1.3 Research Goal

It is possible to formulate the goal, requirements and constrains of a failure prediction concept with the given motivation and goals of the PAHMIR project. Important factors are the aircraft environment and generalization of the concept. This leads to the following goals for the concept:

- An adaptable system condition monitoring
- Simple and verifiable monitoring algorithms
- Failure prediction with 500 flight hours in advance
- Condition monitoring should be online and offline possible
- Condition monitoring and prediction needs to be usable for a changeable cabin layout
Predictive Health Monitoring for Aircraft Systems using Decision Trees

- Low number of needed sensors
- Low hardware profile
- Use new and existing sensors
- Preferable not only limited to the aircraft domain
- Low human interaction

Goal of the research is to develop a condition monitoring and forecasting concept that is usable in the aircraft environment, that can be used for different systems and can be used by operators without much system knowledge and knowledge of the monitoring and prediction concept. It is critical that the developed concept can be used for different systems without much work to adapt it. This is because the system shall be easy to use by different developers for different aircraft systems. Also, to be able to use the concept in the aircraft environment, it is necessary that the used algorithms can be easily verified and understood by a human. This makes it easier to ensure that the system correctly monitors the system condition and also that it forecasts the condition correctly.

The given requirements lead to the concept that is given below. The concept is a software concept that can be embedded in different environments. The largest amount of computation takes place during the configuration of the failure prediction system and not during the operation. Most used computations and methods are fast to calculate and need not much hardware power or memory. The concept needs sensor input and a way to output the predictions.

1.4 Concept

The developed concept for predictive aircraft health monitoring is able to predict failures so that maintenance can be planned ahead. The developed concept is based on two different processes (condition monitoring and condition forecasting) that work together to create a complete concept. The two processes use decision trees (see Section 2.4). Decision trees are used to make decisions in the concept and are the core of both processes. In the first process (condition monitoring) the task is to decide, which condition the sensor data represents and in the second process (condition prediction) the task is to decide how to predict data points. Both tasks are solved by the decision tree.
The core idea behind the concept is to use machine learning to create an expert system. A human expert is needed for configuring the starting parameters and linking sensor data to a system condition during the training. After the training the system is able to work without a human expert. The expert system is designed as a statistical system model to allow a high level of adaptability. A statistical system allows the user to use measurement data to create a system model without the need of full system knowledge. The two processes use parameter optimization to improve the performance of the decision trees and the overall performance. Optimization reduces the need for a human after the initial data and parameter configuration. All process parameters that may change can be changed until an optimal parameter set is found or until a number of different decision trees have been calculated.

The concept can be embedded in most hardware platforms and is system independent. The training of the decision algorithms can be done on any platform and the resulting code is based on simple "if-then-else" statements, which can also be implemented in most platforms. Digital signal processors (DSP) are especially suited for the condition monitoring, because they can calculate the signal processing very fast. With an optimal hardware architecture and a good implementation it is possible to use the condition monitoring and the condition prediction in real-time. Signal processing and prediction parameter approximation parameter calculation take most time.

1.4.1 Condition Monitoring

The condition monitoring concept uses sensor data to calculate the current condition of the system. This can be the system state (e.g. normal, error 1, error 2 ...) or the remaining life time. The concept does not rely on a special type of sensor or only sensor data from one source or kind. It is possible to use any kind of data or combination of data. The concept works best with sensor data which changes multiple times per second. If the data is not numerical, then the preprocessing can not be applied, but it is still possible to use all other parts of the process. This makes it possible to merge data from different sources and of different kinds into one system condition. An extra output of the concept is the similarity of the current sensor data to another system condition and not only to the class it was mapped to. The condition monitoring process is shown in Figure1.3.
Figure 1.3  Condition monitoring process
Condition monitoring is based on a decision tree, which calculates a decision based on signal features. The decision tree needs to be trained with data samples. Preparation of training samples (signal feature extraction) is a complex task and controlled by parameters. The performance and adaptability are improved by an optimization process. Condition monitoring is a simple process compared to the training process. The following methods and technologies are used for the concept:

- Decision trees (Section 2.4)
- Signal analysis (Section 2.2)
- Optimization (Section 2.5)

Fuzzy decision tree evaluation is used and needed to provide also continuous results (percentage values) in addition to the discrete decision of the decision tree evaluation. The continuous results are the similarity of the data belonging to another class. The value of the failure case is used as an input to the condition prediction process, which needs a continuous value.

1.4.2 Condition Prediction

Condition prediction takes a time series (chronological ordered data points) and predicts future data points based on learned patterns/knowledge. It is possible to train the system to predict data points in the close future or in the far future. Prediction is done by calculating a suitable approximation based on learned experience. The condition prediction process is shown in Figure 1.4.

Training of the decision tree is the most complex task of the process like for condition monitoring. The time series, which shall be predicted needs to be prepared and features need to be extracted. Performance is also improved by an optimization process. While the process looks more complicate than the condition monitoring it is easy to compute and the steps are easy to understand by a human. The following methods and technologies are used for the concept:

- Decision trees (Section 2.4)
- Time series analysis (Section 2.3)
- Optimization (Section 2.5)
Figure 1.4  Condition prediction process
1.4.3 Failure Prediction

Failure Prediction combines the condition monitoring and the condition prediction processes into one complete process that can be used to forecast a failure. Base for the failure prediction is the condition monitoring which gives the process the current system state (if correctly trained). However for the failure prediction there is no direct need for the current system state. What is needed is the similarity of the current state to a failure state. This gives the user more info than just the system state, because the a sample might be from the border of a class. Fuzzy decision tree evaluation [4] allows it to take any decision tree and calculate how similar a sample is to another class. A side effect is that the fuzzy evaluation converts the discrete result of the decision tree classification into a continuous number, if you just want to know how similar a sample is to a specific class. For the failure prediction this class is the class of the failure that shall be predicted. Continuous numbers are needed as the input for the condition prediction process, which predicts the trend of the class based on past data.

- Condition Monitoring
- Fuzzy decision tree evaluation
- Condition Prediction
Theoretical Background

2.1 Condition Monitoring

Condition monitoring is part of condition based maintenance\[6\]. Maintenance is the combination of all technical and associated administrative actions intended to retain an item in, or restore it to, a state in which it can perform its required function\[7\]. The goal is to prevent fatal damage for machine, human or environment, to prevent unexpected machine failure, condition based maintenance planning, safety of production and quality control\[8\]. Figure 2.1 shows a breakdown of different maintenance strategies.

Basically there are three different maintenance strategies \[6\][8]:

- **Run-to-break** is the most simple maintenance that is often used for systems that are cheap and where a damage does not cause other failures. The machine or system is used until it breaks. It is commonly used for consumer products\[8\].

- **Preventive Maintenance** is the most common maintenance method for industrial machines and systems. Maintenance is performed in fixed intervals. The intervals are often chose so that only 1-2 % of the machine will have a failure in that time\[6\].

- **Condition-Based Maintenance** is also called predictive maintenance. Maintenance is dynamically planned based on machine or
Figure 2.1  Maintenance[9]

system condition. Condition-Based Maintenance does have advantages compared to the other two strategies, but requires a reliable condition monitoring method [6]

Figure 2.2 shows a typical machine condition based monitoring case. First the machine goes into operation and then it is in normal operation. The machine is replaced short before a failure happens[8].

Condition monitoring can be performed in two strategies for monitoring[6][8]:

- **Permanent monitoring** is based on fix installed measurement systems. Often these systems need to be very complex to react correctly if a failure occurs. These systems are often used if a fast reaction is required after a failure. Permanent monitoring often shuts down a machine if a dangerous failure is detected[6].
Theoretical Background

Figure 2.2 Machine/system condition over time[8]

- **Intermittent monitoring** is generally used to failure prediction and diagnosis. Measurements are taken or regular basis with a mobile device. Data evaluation is done at an external device. Intermittent monitoring is often used to give a long term advance warning[8].

Permanent monitoring is often more easy than intermittent monitoring, because fast reaction times are required. Intermittent monitoring can be more complex and can do more complex computations[6]. Permanent and intermittent monitoring can be combined using the same sensors and working in parallel. This allows the intermittent monitoring to be carried out more often (data is always available[6].

Different methods for condition monitoring are[6]:

- **Vibration analysis** measures the vibration of a machine or system and compares it to a given vibration signature. Vibrations can be linked to events in a machine based on their frequency. Therefore a vibration signal is often analyzed in the time domain and in the frequency domain. Vibration analysis is often used for condition monitoring[6][8].

- **Lubricant/Oil analysis** analyzes the quality of the fluid and if particles are in the fluid. Contaminants in lubrication oils and hydraulic fluids can lead to the failure of the machine/system. The
physical condition of a fluid can be measured in viscosity, water content, acidity and basicity. For a condition monitoring strategy this means condition based oil change. It is also possible to detect wear debris of mechanical systems with a particle analysis[9].

- **Performance analysis** is an effective way of determining whether a machine is functioning correctly. Performance analysis monitors process parameters such as temperature, pressure, flow rate or processed items per hour[6].

- **Thermography** is used to detect hot spots in a system or a machine. At this time thermography is used principally in quasi-static situations.

Condition monitoring can be used for one sensor or for a complex system. Two approaches are used for monitoring a system: one-to-one and one-to-many[9]. In one-to-one monitoring a system parameter which is measured by a sensor is directly forwarded to a signal processing and a condition monitoring (see Figure 2.3) independent of the sub system that the parameter belongs to[9].

![Figure 2.3 One-to-one condition monitoring][9]

In one-to many monitoring one sensor is used to give the condition monitoring information on more than one sub system (see Figure 2.4). One-to-many monitoring helps with failure location[9].

There are different methods for failure detection using condition monitoring. If only one sensor/parameter is evaluated then trend analysis
or limits can be used\cite{8}. Using a \textbf{limit} is to most simple method. The sensor signal is compared to a given limit. A failure has occurred if the sensor signal is greater than the given limit. A limit based failure detection can not be used to predict failure\cite{8}. \textbf{Trend analysis} records a time series of the sensor signal. It can be assumed that the machine operates normal, if only small changes occur over time. A stronger change in the time series indicates the development of a failure. Trend analysis can be used for failure prediction\cite{8}.

If a system is monitored then a system model needs to be created (see Figure 2.5). The model is used to compare the actual system outputs to the theoretical outputs and a difference between both signals indicates an error or a upcoming error\cite{8}. A system can be modeled by a mathematical description through Laplace-based system models or through dynamic (statistical modeling)\cite{9}.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{system_model.png}
\caption{System model\cite{9}}
\end{figure}

The \textbf{mathematical model} tries to describe the system in equations. A mathematical model can become quite complex but a complete definition of the system is often not needed\cite{9}. \textbf{Laplace-based system}
models use the Laplace transformation to model a system with one or more building blocks (See Figure 2.7)\cite{9}. System modeling and simulation tools like MATLAB Simulink, Modelica ... use Laplace like building blocks.

Dynamic fingerprinting works without full knowledge of the system. The output for a given input is recorded and the collection of these records make the model\cite{9}. Outlier detection uses different methods.
Theoretical Background

and techniques to detect an anomaly or a fault in sensor data. Often an outlier stands for a system fault[10].

Other methods for a system modeling and condition monitoring include the use of neural networks and other machine learning techniques. Machine learning and pattern recognition is often used for condition monitoring and trending in complex systems[6][8]. In [11] is an example of such an approach shown. [12] uses a neural network for sensor fusion (one-to-many) while [13] an example of the one-to-many concept for distributed agents is. A real time monitoring with a neural network is shown in [14].

2.2 Signal Analysis

A signal is a vary quantity whose value can be measured and which conveys information[15]. Signals can represent sound, vibrations, color values, temperature ... There are two types of signals: analogue and digital. An analogue signal is a continuous signal and a digital signal does have a finite number of values. The process of transforming an analogue signal into a digital signal is called sampling. Sampling represents an analogue signal by a number of regular spaced measurements or samples[15]. Figure 2.8 shows the sampling of an analogue signal.

The number of regular spaced samples per second is the sampling rate and measured in Hz. A signal does have an amplitude and a phase. The amplitude is the sampling value and the phase measures the time delay this motion and another motion of the same speed[15]. Signals represented as above are in the time domain. It is possible to transform signals so, that they are represented in the frequency domain. In the frequency domain are the signal represented by cosine and sine function with different frequencies[15]. The process which converts the signal is called Fourier transform for analogue signals and discrete Fourier transform for digital signals. Equation 2.1 shows the discrete Fourier transform.

\[
Z(f) = \sum_{k=0}^{N-1} z(k)e^{-2\pi jfkN} \tag{2.1}
\]

\(Z(f)\) is the Fourier coefficient at frequency \(f\)[15]. N is the total number of samples and \(k\) is the current sample. \(z(k)\) is \(x(k) + jy(k)\), where \(x\) and \(y\) are the amplitude and the phase of the signal. It is also possible
to reverse the transform by used Equation 2.2.

\[ z(k) = \frac{1}{N} \sum_{f=0}^{N-1} Z(f)e^{2\pi jfkN} \]  

(2.2)

It is also possible to treat the complex values as real values if the phase is unknown or zero. In this case of an only real valued signal only \( N/2 \) coefficients are independent. This is because \( Z(N - f) \) and \( Z(f) \) are the same if only the real part is considered. For practice this means that \( 2N \) samples are needed to get \( N \) Fourier coefficients. Figure 2.9 shows a real valued signal transformed into the frequency domain.

An algorithm to compute the discrete Fourier transform on a computer is called the fast Fourier transform (FFT). The FFT requires that \( N \) is a power of two[15].
A filter is a process which changes the shape of a signal[15]. Often filters change the signal in the frequency domain. Usual types of filters are low-pass, high-pass or band-pass filters. Low-pass filters keeps low frequency components of the signal and blocks high frequency components. A high-pass filter blocks low frequencies and keeps high frequencies. A band-pass filter blocks all but a given range of frequencies[15]. One way to apply a filter is to transform the time domain signal into the frequency domain, apply the filter and the transform the signal back into the time domain.

Band-pass filters can be used to extract frequency components from a signal into a new signal. If multiple band-pass filters are applied to a signal to extract different frequencies then the filter is called filter bank. The individual band-pass filters can either have the same size or the size can vary[16]. Figure 2.10 shows a filter bank with equal sized band-pass filters and Figure 2.11 shows a filter bank with band-pass filters of a
2.3 Trend Series Analysis & Forecasting

A time series is a chronological sequence of observations on a particular variable\cite{17}. This means that a time series is a number of data/time pairs that are ordered chronological. Figure 2.12 shows some time series. Time series analysis is done to discover historical patterns, which can be used for forecasting\cite{17}. Forecasting is defined as: Predictions of future events and conditions are called forecasts, and the act of making such a prediction is called forecasting\cite{17}. Goal of forecasting is to reduce the risk of decision making\cite{18}.

Time series analysis and forecasting are used in many different areas from economic forecasting and logistic management to strategic management\cite{17}\cite{19}\cite{18}. A time series is defined by\cite{17}\cite{18}:

- Trend is the upward or downward movement of a time series over a period of time.
- Cycle refers to recurring up and down movements around trend levels.
Seasonal variations are periodic patterns that complete themselves in a calendar year.

Irregular fluctuations are movements that follow no patterns.

Time series can be split up into two categories: continuous and discrete. A continuous time series is recorded at all the time, while a discrete time series is recorded at given intervals (hourly, daily ...)[19]. Time series forecasting can be influenced by many factors like the availability of data, cost of analysis or management preferences[18]. Forecasting is defined by[18]:

- Forecasting period is the basic unit of time for which forecasts are made.
made (hours, days, weeks ...).

- Forecasting horizon is the number of periods in the future covered by the forecast.

- Forecasting interval is the frequency in which forecasts are made.

Often the forecasting interval is the same as the forecasting period. This means that the forecasting is revised after each period[18]. Two types of forecasts can be made: expected value in the future and prediction interval[18][17]. The prediction interval is an interval in that has a stated chance of containing the future value. Usually two forecasting strategies are available: qualitative and quantitative methods[17][18]. Qualitative methods involve an expert while quantitative methods analyze the historical observations to predict the future. Bases for the forecasting is to develop a model of the historical data. The model can be based on a single time series (uni-variant model) or it can include multiple variables (causal model)[17][19][18]. Figure 2.13 shows a simple sample model of a time series.

**Figure 2.13**  A linear model of a time series[18]

Different methods are available for quantitative forecasting[17][19][18]:

- Simple Linear Regression
Theoretical Background

• Multiple Regression
• Moving Average Model
• Exponential Smoothing
• Box-Jenkins

Each of the five methods will be explained in this section. Simple linear regression and multiple regression methods can be used to calculate a trend in a time series[17].

2.3.1 Simple Linear Regression

The most simple regression method is simple linear regression. Goal of the simple linear regression is to model the time series with a single straight line (Figure 2.13)[17][18]. The model does have two parameters: the slope and the y-intercept. The model can be written as:

\[ y = b_0 + b_1 x + \epsilon \]  \hspace{1cm} (2.3)

A usual method to estimate the two parameters \( b_0 \) and \( b_1 \) is to use least-squares[17][18]. Least-squares tries to find parameters for which the error sum of squares is the least. This means the sum of the squared error between the line and the point \( y_i \) The error sum can be written as:

\[ \ell(b_0, b_1) = \sum_{i=1}^{n}(y_i - b_0 - b_1 x_i)^2 \] \hspace{1cm} (2.4)

The complete equation for the calculation of \( b_0 \) and \( b_1 \) is[17][18]:

\[ b_1 = \frac{n \sum_{i=1}^{n} x_i y_i - \left( \sum_{i=1}^{n} x_i \right) \left( \sum_{i=1}^{n} y_i \right)}{n \sum_{i=1}^{n} x_i^2 - \left( \sum_{i=1}^{n} x_i \right)^2} \] \hspace{1cm} (2.5)

and \( b_0 = \bar{y} - b_1 \bar{x} \)

where \( \bar{y} = \frac{\sum_{i=1}^{n} y_i}{n} \) and \( \bar{x} = \frac{\sum_{i=1}^{n} x_i}{n} \)

The fitted simple linear regression model is:

\[ \hat{y} = b_0 + b_1 z \] \hspace{1cm} (2.6)
2.3.2 Multiple Regression

Multiple regression is similar to simple linear regression, but the regression depends on more than one variable (see Equation 2.7)\cite{17} \cite{18}.

\[ y = b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_n x_n + \epsilon \]  
\(2.7\)

The variables \(x_1, \ldots, x_n\) can be different functions of time like \(x_1 = x^2\)\cite{18}. \(x_1, \ldots, x_n\) may also be other time series like temperature and sales which may influence a time series. Equation 2.8 shows an example.

\[ y = b_0 + b_1 w(t) + b_2 s(t) \]  
\(2.8\)

where \(w(t)\) is a function over time like the weight of a human over time and \(s(t)\) is also a function over time like the salary. 2\textsuperscript{nd} order or higher order polynomial models can be used\cite{17}. Figure 2.14 shows some 2\textsuperscript{nd} order functions.

![Figure 2.14 2\textsuperscript{nd} order polynomial models \(y = b_0 + b_1 x + b_2 x^2\)\cite{17}]

The general representation of \(p\textsuperscript{th}\) order polynomial model is:

\[ y = b_0 + b_1 x + b_2 x^2 + b_3 x^3 + \cdots + b_p x^p + \epsilon \]  
\(2.9\)

Multiple regression also uses the least-squares method to calculate the parameters \(b_0, \ldots, b_n\). The least squares problem is often describes in
a matrix form like in Equation 2.11[18]. The problem as a matrix is defined as[18]:

\[
\hat{y} = Z\hat{b}
\]  

(2.10)

\[
\begin{pmatrix}
13 \\
20 \\
5
\end{pmatrix} =
\begin{pmatrix}
3 & 2 \\
12 & 4 \\
19 & 34
\end{pmatrix}
\begin{pmatrix}
b_0 \\
b_1
\end{pmatrix}
\]  

(2.11)

The normal equations can be used to solve the least-squares problem in a simple way (Equation 2.12)[18].

\[
\hat{b} = (Z'Z)^{-1}Z'y
\]  

(2.12)

Normal equations is not the most stable method to solve the problem. The QR factorization solves the problem in a more stable way[20].

### 2.3.3 Moving Average Model

The moving average model can be seen as a more simple form of the simple linear regression model. Not the complete time series is evaluated, but only N points of the time series[18]. The moving average model can be seen as a way to reduce the noise in a time series. In the most simple case is the \( y_i \) the average (arithmetic mean) of the last N values[18]. The equation for the moving average model is:

The moving average model can also be used to forecast a trend by using the following equation[18].:

\[
M_\tau = \frac{y_\tau + y_{\tau-1} + y_{\tau-2} + \cdots + y_{\tau-N+1}}{N}
\]  

(2.13)

The equation calculates a forecast of \( \tau \) periods into the future[18].

\[
\hat{y}_{T+\tau}(T) = 2M_T - M_T^{[2]} + \tau \left( \frac{2}{N-1} \right) (M_T - M_T^{[2]})
\]  

(2.14)

where \( M_T^{[2]} \) is a second-order statistic (moving average of the moving averages):

\[
M_T^{[2]} = \frac{M_T + M_{T-1} + \cdots + M_{T-N+1}}{N}
\]  

(2.15)
2.3.4 Exponential Smoothing

Exponential smoothing is a method for smoothing similar to the moving average model. The difference is that the data points are weighted unequal. The most recent data point is weighted more than past data points[17][18]. The equation for a simple exponential smoothing is:

\[ S_T = \alpha x_T + (1 - \alpha)S_{T-1} \]  

(2.16)

\( S_T \) is a weighted average of all past observations. To define an exponential smoothing that is an n-period moving average \( \alpha \) is set to[18]:

\[ \alpha = \frac{2}{N + 1} \]  

(2.17)

The starting value \( S_0 \) can be gained by taking the average of certain number of past data points or to choose it[17][18]. The forecast for the time period \( T+1 \) is \( S_T \)[18]. A low value of \( \alpha \) causes the forecast to weight the last value more and makes the forecast react faster to changes but also to noise. A low values of \( \alpha \) lets the forecast react more slowly.

2.3.5 Box-Jenkins

The Box-Jenkins methodology was developed by Box and Jenkins in 1976. The methodology consists of a four step iterative procedure[17]:

1. Tentative identification: historical data are used to tentatively find an appropriate Box-Jenkins model.
2. Estimations: historical data are used to estimate the parameters of the tentatively identified model.
3. Diagnostic checking: various diagnostics are used to check the adequacy of the tentatively identified model and, if need be, to suggest an improved model, which is then regarded as a new tentatively identified model.
4. Forecasting: once a final model is obtained, it is used to forecast future time series values.

Box-Jenkins models are the autoregressive model, moving averages model, autoregressive-moving average model and autoregressive integrated moving average mode. **Autoregressive processes** use weighted
past data to predict a future value. A white noise signal (fixed variance and mean zero) with a defined variance (that is the same for each period $t$) and is added to the past data\[19\]. The autoregressive process is defined as\[18\]:

$$x_t = \xi + \phi_1 x_{t-1} + \phi_2 x_{t-2} + \cdots + \phi_p x_{t-p} + \epsilon_t \quad (2.18)$$

where $\epsilon_t$ is white noise, $\xi$ is a constant and $\phi_1, \ldots, \phi_p$ are parameters (weights) of the model. The random shock $\epsilon_t$ describes the effect of all other factors than $x_{t-1}, \ldots, x_{t-p}$ on $x_t$\[17\]. An autoregressive process of the order $p$ is call AR($p$)\[18\]. Where $p$ is the number of past data points. Autoregressive processes use the fact that the value of the time series are correlated\[17\]. Related to the autoregressive processes are the moving average processes. The moving average process is defined as\[18\]:

$$x_t = \mu + \epsilon_t - \theta_1 \epsilon_{t-1} - \theta_2 \epsilon_{t-2} - \cdots - \theta_q \epsilon_{t-q} \quad (2.19)$$

where $\mu$ is the mean of the time series, $\epsilon_t, \ldots, \epsilon_{t-q}$ are random shocks and $\theta_1, \ldots, \theta_q$ are a finite set of weights. A moving average process of order $q$ is called MA($q$). The random shocks for the moving average process are also white-noise random shocks, this means that they have the mean zero, have a normal distribution and are defined by the variance. It is possible to combine the autoregressive process and the moving average process. The combined model is called autoregressive-moving average (ARMA) and does have two orders $p,q$ or ARMA($p,q$)\[18\]. ARMA models can only represent stationary time series\[18\]. A time series is stationary if the statistical properties (for example mean and variance) of the time series are essentially constant through time\[17\].

It is possible to convert a non-stationary process into a stationary process by calculating the differences between two successive values. The first differences of the time series values $y_1, y_2, \ldots, y_n$ are\[17\]:

$$z_t = y_t - y_{t-1} \quad (2.20)$$

where $t = 2, \ldots, n$

Taking only one difference is called first differences. If the first differences is also no stationary process then it is possible to take again the differences and get a second differencing. Figure 2.15 shows a second differing\[18\].
Figure 2.15  Differencing a time series two times[18]
**Theoretical Background**

Autoregressive integrated moving average (ARIMA) models use the $d^{th}$ difference to model non-stationary time series. An ARIMA model does have an order of $(p,d,q)$ where $d$ is the $d^{th}$ difference of the original series [18].

### 2.3.6 Other methods

There are many different approaches available for modeling and forecasting a time series. [21] uses an artificial neural network to forecast stock prices. [22] uses piecewise linear approximation to detect trend in a streaming time series. Bayesian and other probability methods are also used to model and forecast a time series [23] [18].

### 2.4 Decision Trees

A decision tree is a tool from the artificial intelligence area. A decision tree is a tree which classifies instances by sorting them down from the root to some leaf node [24]. Each node in the tree specifies a test on an attribute and each branch from to node to another node or leaf corresponds to a test result [24]. An example decision tree is shown in Figure 2.16. The example decision tree classifies the weather if it is suitable to play tennis or not.

![Decision tree example](image)

Figure 2.16 *Decision tree example* [24]

If the decision tree is used to learn a discrete valued function (like the example) it performs a classification. If the tree is used to learn a
Predictive Health Monitoring for Aircraft Systems using Decision Trees

continuous function it performs a regression[25]. Any decision tree can be converted into a logical expression[25]. The example can be expressed as:

\[(Outlook = sunny \land Humidity = normal)\]
\[\lor (Outlook = overcast)\]
\[\lor (Outlook = rain \land Wind = weak)\]

(2.21)

An instance to test consist of attribute value pairs. Each instance is described by a fixed set of attributes (e.g. Outlook) and their values (e.g. Sunny). Decision tree learning is based on a number of provided samples which specify the problem. The set of examples is called training set. There are different algorithms to learn a decision tree. The basic decision tree learning algorithm works as followed[25].

1. Create a new node
2. Split the examples based on the values of the best attribute for splitting.
3. Check for each value of the attribute:
   (a) If the remaining examples have a different classification, then choose the best attribute to split them and create a new child node.
   (b) If all remaining examples have the same classification then the tree is trained. It is possible to make a final classification. Create a leaf.
   (c) If there are no examples left, it mean that no such example has been observed.

There is an error in the training examples, if two or more examples have the same attribute values but different classifications. In this case it is possible to return the classification of the majority of the classifications or to report the probability for each classification[25]. A common method for selecting the best attribute to split the examples is the ID3 and the C4.5 by Quinlain[24]. The idea of ID3 is to select a node based on the information gain. Information needs to be defined to define information gain and understand the concepts. Information entropy is the knowledge that is contained in an answer depending on one's prior knowledge. The less is known, the more information is provided. In
Theoretical Background

Information theory information entropy is measured in bits. One bit of information entropy is enough to answer a yes/no question about which one has no data[25]. The information entropy is also called information and is calculated as shown below in Equation 2.22. $P(v_i)$ is the probability of the answer $v_i$.

$$I(P(v_1), \ldots, P(v_n)) = \sum_{i=1}^{n} -P(v_i) \log_2 P(v_i)$$  (2.22)

The information gain from an attribute test (setting the value of a node in a tree, see Figure 2.16 for an example) is the difference between the total information entropy requirement (the amount of information entropy that was needed before the test) and the new information entropy requirement. $p$ is the number of positive answers and $n$ is the number of negative answers[25].

$$Gain(X) = I(\frac{p}{p+n}, \frac{n}{p+n}) - \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} \cdot I(\frac{p_i}{p_i + n_i}, \frac{n_i}{p_i + n_i})$$  (2.23)

The performance of a decision tree can be tested with a number of test examples. Test examples are examples from the training data, which were not used for the learning. Performance of the decision tree depends on the number of correct classified examples.

A common problem for decision trees is over fitting if noise is in the training data or when the number of training examples is to small[24]. A model has a bad performance with testing data if it is over fitted. A simple method to remove over fitting is decision tree pruning. Pruning works by preventing recursive splitting on attributes that are not clearly relevant[25]. Pruning means to remove a sub-tree from the decision tree. Information gain can be used name irrelevant attributes. Another possibility to reduced over-fitting is cross-validation. In cross validation multiple decision tree are trained each with a different set of training and testing examples. The decision tree with the best performance is chosen. A K-fold-cross-validation means that $k$ different decision tree are trained and each is tested with a different set $1/k$ of the examples[25].

Decision trees can be extended to handle the following cases[25]:

- Missing data: not all attribute values are known for all examples.
Predictive Health Monitoring for Aircraft Systems using Decision Trees

- Multivalued attributes: The usefulness of an attribute might be low if an attribute does have many different possible values (Name or credit card data).

- Continuous and integer-valued input attributes: numerical attributes often have an infinite number of possible values. A decision tree typically chooses a split point that separates the values into groups (e.g. Weight < 160).

- Continuous-valued output attributes: the tree does have at the leaves a linear function rather than a single value (regression tree).

Another class of decision trees are fuzzy decision trees. Fuzzy decision trees are not based on crisp training data, but on fuzzy training data. [26][27][28] have examples of fuzzy decision tree training and uses of fuzzy decision trees.

2.5 Local search and optimization

Local search is a special area of search algorithms. In many cases the search algorithm does have a memory of the way to the solution. This means the algorithm knows which steps it took. Local search algorithms have no memory and know only the current state. It might be possible that they check a member of the search space twice. Local search algorithms do not search systematically[25]. Hill climbing search (greedy local search), simulated annealing or a genetic algorithm are all local search algorithms.

Local search algorithms can not only be used to find a goal but also for pure optimization problems. Local search algorithms work in a state space landscape (Figure 2.17). Each state does have a corresponding location and the elevation of the state/location is the value of the heuristic cost function. Goal is to find the state/location with the lowest elevation(costs). It is also possible to find the highest peak if the elevation is not the costs[25].

The hill-climbing algorithm is a simple loop that moves in the direction of the increases value. Hill climbing only evaluates the neighbor states and then choses the best of those. For this reason hill-climbing is sometimes called greedy local search. Hill-climbing can get stuck fast, because it makes no downhill moves and stays on a plateau or a local maxima[25].
Simulated annealing is a hill-climbing algorithm that can move downwards. The algorithm is based on the annealing process in metallurgy. The metal gets into a fixed state as it cools down. The simulated annealing algorithm selects a random move and if it improves the situation it is accentuated. If not then the move is accepted based on a probability value. The probability decrease exponential with the of the move. The probability also goes goes with each step\cite{25}.

A genetic algorithm not only keeps one state in memory but more than one. The states in memory are called population. Turing each step new states (individual) are calculated based on the current population. The first population is generated randomly. New individuals are calculated through crossover and mutation. In cross over two individuals are chosen from the population based on their fitness. Then two new individuals are created by taking a part of one parent and another part of the other parent. Thus the new individual is created by having a part of both parents. The second child is constructed out of the not selected parts of both parents. Mutation modifies each individual based on an independent probability. Figure 2.18 shows an example of a genetic algorithm. The children form the new population\cite{25}\cite{24}. \cite{29} use genetic algorithms to adapt approximation functions from old problems to new problems and \cite{30} use genetic algorithms to select features for decision trees.
2.6 Literature Review

This subsection takes an evaluation of different publications that had an influence on this work.

2.6.1 Look-Ahead Based Fuzzy Decision Tree Induction

Look-Ahead based Fuzzy decision tree induction is a new method to create a decision tree. The method does not use the greed top-down method, but uses a look-ahead method. A look ahead decision tree induction takes a node and calculates the split. Then each branch is again evaluated. Thus the algorithm is looking ahead. However Look-Ahead algorithms don’t always produce the best results. The paper offers a method to increase the results of Look-Ahead algorithms and proves this with experimental results[28].

2.6.2 Fuzzy Decision Tree for Data Mining of Time Series Stock Market Databases

The paper proposes a method for a new fuzzy decision tree to classify stock trading time series. Example stock market data is used through the paper to show example of how to construct the fuzzy decision tree. However the experimental result section is very short[27].

2.6.3 Induction of Decision Trees

This paper describes the ID3 decision tree induction algorithm by Quinlan. ID3 decision trees can deal with incomplete or noisy data. Quinlan explains his ID3 algorithm and uses many examples to show how the algorithm works. The paper concludes with a conclusion. This paper
Theoretical Background

was the bases for the decision tree algorithms that were used in this thesis and the experiments[31].

2.6.4 A complete fuzzy decision tree technique

This paper presents a method call "soft decision trees" (SDT). Soft decision trees output a numerical value instead of a crisp decision and share similarities with regression trees. Soft decision calculate the membership of an object to a class, similar like the fuzzy decision tree evaluation concept in this thesis. Soft decision trees however use fuzzy input values[26].

2.6.5 Intelligent stock trading system by turning point confirming and probabilistic reasoning

This paper talks about the application of probabilistic models for stock market systems. A Markov Network is used to detect turning points in the stock data based on different indicators. The paper shows the usability of the concept with experiments using real world data[23].

2.6.6 Continuous Trend-Based Classification of Streaming Time Series

Streaming time series are time series where new values arrive and a new trend has to be calculated every time. This is the typically the case for online monitoring. The paper tries to classify trends for streaming time series. An algorithm is given and the performance is analyzed using stock market data and Tropical Atmosphere Ocean data. Classifying streaming time series was also a problem in this thesis[22].

2.6.7 A Neural Stock Price Predictor using Quantitative Data

The paper describes the usage of a neural network to predict stock market prices. In addition to only calculating the prediction the method in this paper also calculates the accuracy of the forecast[21].

2.6.8 A Neural Network Approach to Condition Based Maintenance: Case Study of Airport Ground Transportation Vehicles

This paper describes a concept for monitoring different types of automated vehicle doors. A neural network is used to learn the character-
istics of the different types of doors and to predict the condition. The concept was evaluated using real world data in an experiment. Three different types of neural network paradigms were evaluated for the used neural network. The paper closes with an interesting discussion about the learned lessons. The topic of the research and the approach for the condition monitoring were similar to the approach taken in this thesis[14].

2.6.9 A Multi-Agent Fault Detection System for Wind Turbine Defect Recognition and Diagnosis

Anomaly detection and data trending are used to detect faults in a distributed environment. The paper proposes an architecture that uses multiple agents to detect a system state. The application does have many similarities to the application in this thesis. Only the focus is on detecting and not on forecasting failures[13].

2.6.10 Sensor fusion of a railway bridge load test using neural networks

This publication uses neural networks to detect the condition of a bridge. An neural network is used to model the input and output relation of finite elements model. The bridge was monitored by multiple sensors. Data from those sensors was fused in the neural network using the data as input to the network. The results of the paper were that a neural network can be used to represent a finite elements model under certain constrains and thus can be used to monitor structures[12].

2.6.11 Automated Rule Extraction for Engine Vibration Analysis

Health monitoring and detection and classification of possible failures is the topic of this paper. The goal is to develop an automatic system for extracting rules for health monitoring using evolutionary programming. Neural networks and evolutionary programming are used to extract rules. With the rules a decision tree is then constructed[11].

2.6.12 A survey of outlier detection methodologies

This publication describes and evaluates different outlier detection methods. It starts with an introduction to the topic of outlier detection and
defining the different types of outliers. The section is followed with a list and description of different methods beginning with statistical methods and parametric methods. The publication also includes the discussion of using neural networks and decision trees for outlier detection. Each method contains a note for what kind of outlier it is usable[10].

2.6.13 Using Genetic Algorithms for Adapting Approximation Functions

This short paper explains how genetic algorithms can be used to modify the parameters of an approximation problem or for selecting a new approximation function[29].

2.6.14 Decision tree classifier for network intrusion detection with GA-based feature selection

This paper proposes a method for feature selection for a decision tree based on a genetic algorithm. The decision tree is then used for intrusion detection of computer systems. The method used in this paper is very similar to the one used in this thesis however it is slightly differently used for a different application[30].
Predictive Health Monitoring for Aircraft Systems using Decision Trees
The condition monitoring process consists of two sub-processes: a training process and a decision process. In the training process the decision tree for the given problem is created. The decision process uses the created decision tree to decide in which condition the system currently is. Goal of the decision monitoring process is to have an adaptable process that can be used to calculate the current condition of a monitored system. This is done with a decision tree trained on the problem. Figure 3.1 shows an activity diagram of the process.

The left side of the diagram shows the decision process and the right side shows the training process. When a new data sample is available it is checked, if it is a training sample (no decision tree yet created) or if it should be evaluated with the decision tree. The process is simple. If the data sample should be evaluated, the data is preprocessed based on parameters and the processed data is then evaluated with the decision tree. Output is a system condition. If the new data is training data then it is labeled/classified and added to the sample database. The decision tree is trained when enough training samples are available. Samples are divided into training and testing samples. The training and testing samples are preprocessed based on initial selected parameters. The decision tree is calculated using any decision tree algorithm e.g. ID3 or C4.5 and the training samples.

The decision tree is tested with the testing sample set. If the performance of the decision tree is below a given limit, then the preprocessing parameters are changed, based on a heuristic optimization algorithm.
Figure 3.1  *Condition monitoring process*
This algorithm can be e.g. Greedy Search, Simulated Annealing or Genetic Algorithm. A new decision tree is calculated based on the modified parameters. The new decision is again tested and compared against a limit. This optimization process is repeated until a given number of decision trees have been calculated or until a decision tree has a better performance than the given limit. The decision tree and the data preprocessing parameters are saved.

Evaluation of a new data sample is a simple task. First the data is processed depending on the preprocessing parameters from the training process. After the data sample is preprocessed it is inputted into the decision tree. Evaluation can be done in two ways: usual decision tree induction or fuzzy decision tree evaluation[4]). Multiple decision trees with different preprocessing parameters can be calculated and used to improve the performance and reduce the noise sensitivity. If multiple decision trees are used then the condition that the majority of the decision trees select is taken.

3.1 Data Requirements

To use the proposed concept a few requirements need to be fulfilled.

**Stable System** A stable system behavior is needed to use the condition prediction. This means that the system should not change its condition frequently. Reason for this restriction is that the condition should not change during the sampling of samples. If the condition changes during the recording then the classifier cannot classify the current state correctly.

**Continuous Sensor Values** The condition monitoring was not developed with discreet values and events in mind. The system needs values which change over time. Most physical system fit these criteria unless their operation is triggered by a highly unpredictable external source. E.g. accelerating a car is a continuous time series, while starting the car or closing a door is not.

**High Frequency Sensor Data** The system works best if the sensor input changes frequently during sampling of a sample (e.g. sound, vibration, power consumption ...). Most data preprocessing steps are developed to work with a signal input with more than 1 Hz. But it is possible to work also with slow changing values like temperature, even if less information can be extracted from such data.

**Periodical Data Sampling** The concept relies on data samples
which are collected during given intervals and not at discrete or random times. This requirement is less for the condition monitoring needed but it is required for the trend prediction to be able to make a forecast. No time information is saved in the recorded data.

**Discrete Data Sampling** Condition monitoring and condition prediction work with fixed sample lengths. This means that it is not possible to record continuous data samples without modifying the algorithm and splitting the continuous data sample into multiple one second samples.

**No True Real Time Monitoring** Calculation of a condition needs some time. The calculation of one condition may take as much as ten seconds, depending on the number of the used decision trees and the number of the sensors. If only one tree is used and only a few FFTs need to be calculated then it is possible to have a calculation time of less than one second. This was the usual case during the experiments. If the calculation time is less than one second then it is possible to calculate the condition while a new data sample is recorded.

**Multiple Conditions** Data samples of more than one condition are needed for learning patterns and classification. With the concept it is not possible to have a "One-Class" classifier. A One-Class classifier is a classifier, which detects incorrect states and conditions based only on the data of the usual operation. Condition monitoring always needs at least data of two different operation modes or conditions. It is possible to change the concept to find incorrect conditions, but this would require a significant change.

The following subsections detail each step of the process.

### 3.2 Training Process

The training process explained in this section calculates one optimized decision tree and a set of optimal preprocessing parameters. The process (except data sampling and labeling) needs to be executed multiple times if more than one decision tree needs to be calculated.

#### 3.2.1 Data samples

The condition monitoring concept uses model based on statistical data to classify new data. Therefore a lot data samples are needed, before the "real" training process can be started. Multiple samples for each different class of the system are needed. How much training data is needed is not
easy to say. It depends on the complexity of the system[3]. Getting enough data and useful data is a difficult process. For new systems it is possible to collect data during system testing and prototyping. Data collecting for older systems is possible if multiple systems are available and data recording can be performed in parallel.

System data can come from many different sources. These sources can be from internal sensors and/or external sensors. In the experiments (Section 6) external sensors were used. The sensor signals were recorded for one second each minute. Different time intervals are possible depending on the dynamic of the system. If the system changes fast then a higher sample frequency is needed. If the system does have a slow dynamic, then a lower sampling frequency can be used.

The proposed concept can work with any kind of input data; however it is assumed that a data sample is a discrete signal with more than one data point. A sample length of one second is enough for most system to extract information, longer sampling periods allow the calculation of frequencies smaller than 1 Hz. For most use cases it is enough to get one data sample every ten minutes during operation or cruise flight. A sensor sampling frequency of higher than 100 Hz is recommended. If a lower frequency is used, then the preprocessing needs to be adapted to that frequency. The signal source does not matter, it can be sound, vibration, temperature, power consumption, weight or magnetic flow data as long as it is a one-dimensional time series source (see Figure 3.2). If more than one data source is used or a data sample does have more than one dimension, then the preprocessing algorithm also needs to be adapted or the data needs to be transformed. The simplest way is to have one preprocessing step for each dimension of the data and then concatenate the preprocessed data before giving it to the pattern recognition. Each used data sample needs to be classified/labeled by a supervisor.

3.2.2 Data Labeling

An important part of the training process is to classify each data sample. The learning algorithm needs all training samples to have a number of features and a classification. The classification of the data samples should describe the condition of the system that the data sample stands for. Some possible classifications for sensor data samples can be:

**Life time** Each system does have a life time after which it needs to be replaced. Life time can be measured in operating hours. If life time
should be used as a condition it is often useful to use the passed life time or the remaining life time of the system. Life time should be represented as a percentage value or in blocks to prevent too many different classes. More classes slow the training and make the system more sensible to noise (over-fitting)[31].

**System mode** System mode means if the system is in normal operation or if some failure occurred. This classification is useful to detect failures in a system.

A good classification can influence the performance of the condition monitoring significantly. Many or very specific classes may cause over-fitting and makes the system sensible to noise.

### 3.2.3 Preprocessing

Signal analysis and machine learning are used to detect the condition of the system. For learning and classification the data samples need to be prepared[2]. The process depends on different parameters. Each of these parameters needs to be adapted to the data. In this concept the selection of the optimal parameters is performed by a genetic algorithm (Section 2.5). These parameters include:

- Signal transformation from the time domain into the frequency domain
- Noise reduction
- Grouping of frequencies
Condition Monitoring

- Calculation of the maximum and mean frequency power of every frequency group
- Calculation of the number of peaks of all groups
- Transformation of the frequency groups back into the time domain
- Calculation of the maximum and mean amplitudes
- Calculation of the maximum and mean values of the complete signal

Noise and the amount of data are reduced and extra information is added during preprocessing to the data. First, the data is transformed into the frequency domain, where the noise is reduced. Then, frequencies are grouped. It is possible that the frequency span of the groups overlap each other. E.g. if the frequencies 1 to 50 belong to one group and have an overlap of 50%, then the second group contains the frequencies from 26 to 75 and the third group contains the frequencies from 51 to 100. Mean and maximum powers are calculated for each frequency group, as well as the number of peaks. Then each group is transformed back into the time domain, where the mean and maximum amplitudes are calculated. The mean and maximum frequency power and mean and maximum amplitude of the complete signal is calculated as a last step. Table 3.1 shows the parameters of the preprocessing and the possible values. Figure 3.3 shows the preprocessing steps.

![Signal preprocessing](image)

**Figure 3.3** Signal preprocessing

- Fast Fourier Transformation
- Noise Reduction
Predictive Health Monitoring for Aircraft Systems using Decision Trees

- Frequency Grouping
- Calculate Group Mean/Max
- Calculate Group Peaks
- Calculate Global Min/Max
- Group Inverse Fast Fourier Transformation: Each frequency group is separately transformed back into the time domain. With this transformation it is possible to analyze the individual groups or frequencies in the time domain, without all other frequencies in the signal.
- Calculate Group Mean/Max
- Calculate Global Mean/Max
- Output: The output of the algorithm are the mean and maximum values of the frequency groups in the time and frequency domain, the number of peaks and the mean and maximum values of the complete signal in the time and frequency domain. This is much less data than pure signal data. The total number of the values depends on the width of the frequency groups (blocks).

Data samples can usually be divided up into two categories of data: high frequency data and low frequency data. Low frequency data is defined as data with a sampling frequency less than 1 kHz. High frequency data is any data with a higher sampling rate than 1 kHz.

The low frequency data will not be processed besides bringing the data into the correct data format for the algorithm. There is too little data to do frequency analysis and compression.

The high frequency data will be processed with the following steps: First the data is transformed into the frequency domain and then noise reduction is applied to the data, after that the frequency data is partitioned into small blocks and finally each block group is enhanced with extra information.

**Fast Fourier Transformation and Grouping** The fast Fourier transformation takes a number of time-domain samples and transforms them into the frequency domain. Basis of the FFT algorithm is the discrete Fourier transformation (see Section 2.2 for information on the Fourier transformation). A fast Fourier transformation is performed in
Table 3.1 Preprocessing parameters

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Possible Values</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Block width</td>
<td>0–1000</td>
<td>Hz</td>
</tr>
<tr>
<td>Block overlap</td>
<td>0–50</td>
<td>%</td>
</tr>
<tr>
<td>Noise reduction</td>
<td>0–5</td>
<td>–</td>
</tr>
<tr>
<td>Calculate the mean amplitude for each block</td>
<td>boolean</td>
<td>–</td>
</tr>
<tr>
<td>Calculate the maximum amplitude for each block</td>
<td>boolean</td>
<td>–</td>
</tr>
<tr>
<td>Calculate the mean frequency power for each block</td>
<td>boolean</td>
<td>–</td>
</tr>
<tr>
<td>Calculate the maximum frequency power for each block</td>
<td>boolean</td>
<td>–</td>
</tr>
<tr>
<td>Calculation the number of peaks for each block</td>
<td>boolean</td>
<td>–</td>
</tr>
<tr>
<td>Minimum Value of a peak</td>
<td>0–5</td>
<td>–</td>
</tr>
<tr>
<td>Calculate the overall mean and maximum values</td>
<td>boolean</td>
<td>–</td>
</tr>
</tbody>
</table>

$O(N \log N)$ operations. A full transformation with the sampling frequency is done. After the fast Fourier transformation is done, the frequencies are divided up into blocks. A frequency group is called a "block". It is possible that frequency groups overlap each other. That means if a frequency group is from 1 to 100 and the overlap is 50%, then the next frequency groups is from 51 to 150 and the following frequency group is from 101 to 200. If the overlap is 0% then the first block is from 1 to 100, the second from 101 to 200 and the third from 201 to 300. The overlap is controlled by the block overlap parameter. The number of the frequencies that are grouped in one block is determined by the calculation parameter Block Width. If less than Block Width frequencies are available, then all frequencies are treated as one block. After partitioning all blocks are transformed back into the time domain, to get information about the behavior of the block-signal over the time. Figure 3.4 shows how a signal in the frequency domain is separated into blocks and how they are transformed back.

Noise Reduction Noise reduction is applied to the signal to remove random data from the samples to improve the feature detection of the
undisturbed signal. The maximum frequency power is calculated and then each frequency signal that is below a given fraction of the maximum frequency power is reduced to zero to remove noise from the sample. The exact fraction of the maximum frequency power for noise reduction is a parameter of the experiments (Noise Reduction Factor).

**Additional Information and Data Compression** Each block of the sampled data is enhanced with extra information. This information is added to give the following algorithm more information about the signal in the time and the frequency domain. The added information is for the time domain:

- The maximum amplitude of each block
- The mean amplitude of each block
- The maximum amplitude of the complete signal
- The mean amplitude of the complete signal

In the frequency domain the following information is added:

- The mean frequency power of each block
The maximum frequency power of each block

The frequency with the highest power of each block

The number of peaks that are higher than a given magnitude of the mean frequency power

The mean frequency power of the complete signal

The maximum frequency power of the complete signal

The extra information is also calculated for the complete signal sample. Experiments showed that the added information is more useful for the algorithm than the raw data. This allows compressing the data. For example, the information of 100 frequencies is reduced down to four attributes (maximum and mean power, the frequency with the maximum power and the number of peaks). Almost the same result is achieved in the time domain. Instead of calculating the amplitude for each frequency in the time domain, only two attributes (maximum and mean amplitude) are calculated for 100 frequencies.

\[
Freq_{\text{Info}} = 4 \cdot \frac{\text{Frequencies}}{\text{BlockWidth}}
\]

\[
Time_{\text{Info}} = 2 \cdot \frac{\text{Frequencies}}{\text{BlockWidth}}
\]

\[
Total_{\text{Info}} = Freq_{\text{Info}} + Time_{\text{Info}} = 6 \cdot \frac{\text{Frequencies}}{\text{BlockWidth}}
\]

\[
Normal_{\text{Info}} = 2 \cdot \frac{\text{Frequencies}}{\text{BlockWidth}}
\]

\[
Compression = \frac{Total_{\text{Info}}}{Normal_{\text{Info}}} = \frac{3}{\text{BlockWidth}}
\]

The needed data is reduced to 3 % if \(\text{BlockWidth} = 100\) and \(\text{Frequencies} = 11000\).

### 3.2.4 Calculate Decision Tree

The sensor data samples are converted to training samples in the preprocessing step. All training samples now have a number of features and a
class. Decision tree calculation uses any of the available algorithms (ID3, C4.5, Random Forests, CART, ...). After the decision tree is calculated it needs to be tested and evaluated. If the performance of the decision tree is below a limit depending on the needed accuracy then it is possible to try to improve the performance by modifying the preprocessing parameters.

### 3.2.5 Optimize Decision Tree

The performance of a decision tree can be improved by modifying the preprocessing process[2]. Processing option can be turned on or off and parameters can be changed. It might be unfeasible to calculate the optimum parameter set depending on the number of the options and their possible combination. If one decision tree calculation takes a long time and if the solution space is large, then it is not possible to test possible combination. Instead a heuristic optimization approach is needed. Greedy Search, Simulated Annealing and Genetic Algorithm are the most common heuristic optimization methods. Some methods may be more useful than others depending on the problem. The genetic algorithm does have the advantage that it can be executed in parallel which reduces the overall calculation time.

A decision tree is calculated with each new generated preprocessing parameter set. The optimization continues until a given number of decision trees have been calculated or until a decision tree has a better performance than the limit[3].

### 3.3 Monitoring Process

The monitoring process classifies new sensor data samples. For this task the new sensor data samples are preprocessed with the same parameter set, which was used to calculate the decision tree. The decision tree then evaluates the data sample and classifies it. Output of the process is a classification that is system condition that the sensor data sample stands for.

### 3.4 Summary

This section showed a process for monitoring the condition of a system. The process needs no knowledge about the system beside classified sensor
Condition Monitoring

data samples. The process can adapt itself to different system with signal processing and heuristic optimization. When the decision tree was calculated it can be used to classify new sensor data samples.
The condition prediction process decides on the best prediction method for the current time series and then to predict a certain number of future data points. The process is divided into two sub-processes. One process for problem training and the other sub-process for prediction (Figure 4.1).

The overall process is very similar to that of condition prediction. Training also uses an optimization loop. But the individual process steps are different. In the training process all training samples are classified by the process and not by a human operator. The human operator only defines the maximum number of past data points and how far into the future the process shall predict the time series. The prediction process contains a loop that calculates multiple predictions.
Figure 4.1  Condition prediction process
4.1 Training Process

The training process does have five fundamental steps: data sampling, data classification, data preprocessing, decision tree calculation and prediction testing.

4.1.1 Data Samples

Generation of time series data samples is controlled by prediction constraints. Time series data samples can be created in a static and in a dynamic way. The basic time series data sample generation process is shown in Figure 4.2.

![Time series sample generation](image)

Figure 4.2  Time series sample generation

Multiple time series data sample are generated from one or more time series. A time series data sample is generated by moving a window over the time series. All data points in the window form a time series data sample. The window is shifted by one or more data points after a sample is taken. The number of data points the window is shifted depends on how many training data samples are needed. The static window size
Predictive Health Monitoring for Aircraft Systems using Decision Trees

where $w$ is the window size, $d_p$ is the number of past data points and $d_f$ is the prediction horizon. It is possible to create and mix time series data samples from different time series for the training, if multiple time series from one problem are available.

A **dynamic window** is also possible. In this case the window grows with each step. The window size starts with only a few data points, but grows with each step. This is normally the case when the training data shall represent the time series as it grows and shall include all past data points. A dynamic window can only be used if the possible features (Section 4.1.3) do not include any features that are dependent on the number of data points.

Figure 4.3  **Dynamic window**

A static window can include the complete time series. In this case just
the separation between past data points and future data points for the training data is changed to create samples.

![Dynamic separation diagram](image)

**Figure 4.4** *Dynamic time series separation*

The data classification step needs the past data and the future data points of a training sample. Data preprocessing needs only the past data points.

### 4.1.2 Data Classification

Data classification is used to calculate the best prediction method for the current time series sample. The calculation is done by testing which of the available approximation methods has the least approximation mean square error for the approximated future data points. A constraint is that the approximation/extrapolation can only be calculated for past data points of the training sample and cannot use the data points that are marked as future data points. The reason for this constraint is that the decision tree later will also only have those data points available. This means an approximation/extrapolation is only calculated for the constrained time series sample (only past data points), but the mean square error for the future data points needs to be low. The following methods can be used for the prediction of data points:

- Linear Regression
- Multiple Regression
4.1.3 Data Preprocessing

Data preprocessing transforms a time series data sample into a training data sample by calculating the time series features. Different features for each sample are calculated. Those features plus the classification, which is calculated in a previous step, then form the training data sample. Which features are calculated and how they are calculated depends on the preprocessing parameters. This process step is similar to the preprocessing in the condition monitoring process. The following features are possible:

- Maximum value
- Mean value
- Minimum value
- Gradient

Controlled is the process by the following variable parameters:

- Maximum number of past data points
- Usage of maximum value
- Usage of mean value
- Usage of minimum value
- Usage of gradient
- Usage of other time series if available.

It is possible to use other features and parameters that are not listed if they can be applied to time series. Preprocessing is only applied to the data points that are marked as past data points. The data points are the same that were used for calculating the classification.
4.1.4 Decision Tree Calculation

A decision tree can be calculated after the training data have been calculated. Decision tree calculation can be done with any available algorithm for decision tree creation. The result is a decision tree that decides which method shall be used to predict data points.

Testing the decision tree for prediction is more complex than for condition monitoring. Standard methods cannot be used, because the time series prediction is the goal and not the decision making. The decision tree is tested by calculating the prediction for the original time series that were used to create the time series data samples. For this step the prediction process is executed multiple times. The prediction is calculated for every possible starting point of the original time series. For each prediction the following values are calculated:

- Maximum squared prediction error
- Mean squared prediction error
- Minimum squared prediction error
- Confidence range for a maximum prediction error of 10 %
- Confidence range for a maximum prediction error of 5 %
- Confidence range for a maximum prediction error of 1 %

Confidence range is the forecasting horizon, where the maximum prediction error is below a defined limit. Confidence range is measured as a fraction of the forecasting horizon that should have been predicted. E.g. a forecasting horizon of 10 data points out of a trained prediction horizon of 100 data points would be a confidence range of 0.1. The measurement of the overall performance is:

\[
p_{\text{pred}} = 6 \frac{w_0}{1 + err_{\text{max}}} + \frac{w_1}{1 + err_{\text{mean}}} + \frac{w_2}{1 + err_{\text{min}}} + w_3cr_{10} + w_4cr_5 + w_5cr_1
\]  

(4.2)

where \(w_0, \ldots, w_5\) are weights between 0 and 1. \(err_{\text{max}}, err_{\text{min}}\) and \(err_{\text{mean}}\) are the calculated prediction errors. \(cr_{10}, cr_5\) and \(cr_1\) are the confidence ranges. \(p_{\text{pred}}\) is the prediction performance value. A lower value indicates a better prediction performance.
4.1.5 Optimization

If the performance of the prediction is lower than a limit, an optimization loop is started. The optimization loop works exactly like the optimization loop for the condition monitoring process. A heuristic optimization is used to modify the parameters for the data classification and the data preprocessing. The parameter for the maximum past data points may be not increased past the maximum past data points limit. The number of the future data points to be predicted may not be changed.

4.2 Prediction Process

The time series prediction is an iterative process based on three steps. First is the given time series preprocessed to extract the time series features. In the second step is the decision tree evaluated to select the best prediction method for the given time series. In the next step one or more data points (based on parameters) are calculated. Predicted data points are added to the time series. The two steps are repeated with the new time series containing the new predicted data points.

4.2.1 Data Preprocessing

The first step is to preprocess the data and calculate the features of the time series that shall be predicted. Data preprocessing depends and the windowing of the training data. If a static window and static data separation was used then the preprocessing should use the same number of past data points that was used in the training. If a dynamic window or dynamic data separation was used then this needs also to be considered when choosing which past data points are used. Data preprocessing uses the same parameters that were used for the training of the final decision tree.

4.2.2 Calculate Prediction Method

The best prediction method can be chosen with the decision tree, when the features have been calculated. The decision tree is evaluated using the default decision tree evaluation rules.
4.2.3 Predict data point(s)

Data points are predicted using the selected prediction method and the time series data sample. The prediction method can be used to predict one or more new data points. Normally a number of future data points shall be predicted that is the same as in the training.

Newly predicted data points are added to the time series data sample. If the prediction horizon is not yet reached the process is executed again using the new data points as past data points.

4.3 Summary

The condition prediction process is more complex than the condition monitoring complex. But if the parameters are set then the process works automatically and creates a prediction method for the current problem. The prediction can is optimized for a certain prediction horizon. It is possible to calculate different decision trees and preprocessing parameters to have a short term forecasting and a long term forecasting. Both methods give different information to the user.
Predictive Health Monitoring for Aircraft Systems using Decision Trees
5

Failure Prediction

Failure prediction is the process to predict a failure in the future based on current and past data. The process takes condition monitoring data from monitored system. Data is enhanced with additional information by fuzzy decision tree evaluation[4] and then used for condition prediction to forecast failures in the future. Failure prediction can be considered as a sample application of the condition monitoring and the condition prediction process. Fuzzy Decision Tree Evaluation is the link between the two processes and thus will be explained in the following subsection, followed by the process description itself.

5.1 Fuzzy Decision Tree Evaluation

Fuzzy decision tree evaluation takes a normal decision tree and evaluates all possible path and calculates the similarity of the input data to each class, where the correct class has a similarity of 1 or 100 %. The evaluation of all paths is done by assigning each decision a weight based on the boolean decision. The "true" decision is is given a weight of one and the "false" decision gets a value lower than one and higher than zero. The value of the "false" decision is calculated based on the distance of the data to the "true" border (decision split). Different methods to calculate the distance can be applied based on the problem. During the evaluation the value for each path is calculated by taking the sum of the weights of the path and then divide the sum by the depth of the path (taking the average of the path values). This results in a value for each leaf. It is possible for one class to have multiple leafs, in this case the largest value of all leafs for one class is used as the result for the
class. The advantage of this evaluation is that the decision tree creation algorithm does not to be changed and it can be applied to any decision tree instead of the default decision tree evaluation.

5.2 Goal Setup

Goal of the failure prediction is to predict a failure. The prediction process takes a time series and predicts the future of the time series. Fuzzy decision tree evaluation returns multiple results for each data sample. If the data sample come in a chronological order with the same time between each sample, then we get multiple time series, one for each possible class. The prediction process can only predict one time series at a time. The user of the failure prediction needs to decide what class he wants to predict. A drawback of the fuzzy decision tree evaluation is that the class of the current sample always has a result of 100 %. This means that the it is not possible to use the condition prediction for the no-failure state, if there is only one, because then the time series will have multiple 100 % values in a row, which makes the prediction as it is implemented impossible. The algorithm does not 'know' at which position in time it is. That means a class needs to be monitored that is not the no-failure class. If there is only one failure class then that class is selected for the prediction, otherwise one prediction for each failure class has to be made. Each predictor has to be individually trained, which increases the training time significantly.

5.3 Training Process

The training process for the condition monitoring part of the failure prediction process is the same as for the default condition prediction. The next step is to use the training condition monitoring tree to create a time series for the prediction process. For this are the samples ordered by time and are processed by the fuzzy decision tree evaluation using the trained tree. This results in multiple parallel time series. One or more time series are now selected based on the selected goal setup. The image below shows the training process for the failure prediction.
5.4 Prediction Process

The prediction process for failure prediction is similar to that of condition monitoring and condition prediction. First the current condition of the system is evaluated based on a data sample. Next the fuzzy decision tree evaluation is applied and the output is the input for the condition prediction together with past data. Here it is possible to perform multiple predictions for different target conditions if needed or wanted. For the prediction it is needed to keep a time series for each target condition in memory. The image below shows how the prediction process looks like.

![Figure 5.1 Failure Prediction Training Process](image1)

5.5 Summary

This section described a method how to combine the previously shown two methods. The combined method can be used for a complete failure prediction. Link between the two methods is the fuzzy decision tree evaluation, which allows the transformation of discrete decision results into continuous data. The prediction process takes the data adds it to data in a database and predicts a time series for one or more different target system states.

![Figure 5.2 Failure Prediction Process](image2)
Predictive Health Monitoring for Aircraft Systems using Decision Trees
Experiments were done to evaluate the condition monitoring and prediction concept. Different experiments for each of the two processes were done. A test rig which was built together with Airbus Operation GmbH was used for the condition monitoring experiments. Matlab was used for condition prediction, because it was not possible to use the test rig to generate a time series that would represent the reality.

Both experiments try to simulate the filter clogging of the high pressure air filters of the air conditioning of the A340-600 aircraft. This system was chosen, because it has no active parts and is completely passive and thus difficult to monitor. But the system is connected to fans and air flows through the filters.

6.1 Test Rig

The test rig was built by Airbus Operations GmbH during the PAHMIR project to provide a testing environment. The test rig is built into an aircraft hull and consists mostly of aircraft parts. The goal is to reassemble the real environment as closely as it is possible and needed. The test rig was constructed from the following parts:

- HP Recirculation Fan of the A340-600
- HP Air Filter of the A340-600
- Ducting
- Electronic Vibration Measurement Box
The Electronic Vibration Measurement Box (EVB) is used to record sensor data. The box is equipped with two vibration sensors and two microphones. One of each sensor type is attached to the fan and to the filter housing. The EVB (Figure 6.2) is a hardware developed in PAHMIR. In total ten boxes were produced for different tasks. Design goal was to have a box which can record and store different sensor data for a long time (8 weeks). During the project the EVB was used to record data from different experiments and at the ground test rig. The EVB was also used on an Airbus Operations test flight in Toulouse to record data [32][33].

Parts breakdown:

- Enclosure
- Autonomic electronic board
- 8 AA batteries
- SD card (16 GB) are stored in the enclosure.
- 4 sensors (2 microphones and 2 acceleration sensors)
Experiments

Dimensions:
Sensor: 2200mm (cable)
Enclosure: 105mm x 55mm x 190mm

Figure 6.2 Open EVB

The autonomic box and the SD – card are stored in the enclosure. The battery pack that powers the autonomic sensor box is equipped with overcharge and short circuit protection. It consists of eight Panasonic LR6AD AA primary batteries, which comply with IEC 60086. The EVB contains two internal sensors in addition to the external attachable sensors: a temperature and a pressure sensor. With the pressure sensor it is possible to detect if the aircraft is in cruise flight or not. With the temperature sensor it is possible to detect the temperature setting of the environment, specifically, if the air conditioning is turned to low or high. Up to four external sensors can be attached to the EVB. Sensor data is recorded as a four channel wave file.

The EVB is recorded with a simple configuration file, which is stored on the SD-card. Configuration options are:

- Sampling frequency (default: 48000 Hz)
- Number of thousands of samples that will be recorded each time (default: 48000)
- Number of seconds the device will sleep in between 2 recordings (default: 600 seconds)
• Now much time the sensors take to stabilize at power on (default: 50 milliseconds)

• Gain for each channel (default: 1)

6.2 Condition Monitoring

The experiments in this section shall show the ability of the concept to detect clogging of air filters by using the processes. The condition monitoring experiment simulate the clogging of an air filter with dust. In addition experiments for evaluating the preprocessing, optimization and fuzzy decision tree evaluation were performed. The preprocessing experiments are shown in [2], the optimization experiments are shown in [3] and the fuzzy decision tree experiments are shown in [4].

6.2.1 Setup

Data for the experiments were collected at the ground test rig. Fan and filter vibration and sound data were recorded. During the data collection each filter was polluted with dust (MIL-Spec (quartz) dust). The dust was added in 25 gram steps going from 25 gram up to 200 gram per filter for a total of eight possible classes. The classifier was trained by applying the optimization process with genetic optimization and calculated 10 generations each with 20 members. The starting population is a random parameter list. 15 data samples were used for every class (pollution grade). To test if the classification accuracy can be increase the experiments were performed with a single tree classifier and with a classifier build out of three different decision trees using the decision that was selected by the majority of the classifiers.

6.2.2 Results

In the experiment the goal was to detect how much dust is in the filters. The complete data set was used for the experiment. For test data the training data was arranged by increasing weight. The classification should detect a nine step function. Table 6.1 shows the number of the samples per class. Figure 6.3 shows that the calculation resembles a step function, just a few classes are classified wrongly. With three classifiers (same training data, different parameters) the number of wrong classified classes drops even more (Figure 6.4). Figure 6.5 shows the results
of the optimization. The red line is the average fitness and the green line is the best fitness. It clearly shows that more generation would have given a better performance. The average and maximum fitness of the population steadily increased.

**Table 6.1 Number of detected classes for one and three decision trees**

<table>
<thead>
<tr>
<th>Dust</th>
<th>1 tree</th>
<th>3 trees</th>
</tr>
</thead>
<tbody>
<tr>
<td>25 gram</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>50 gram</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>75 gram</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>100 gram</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>125 gram</td>
<td>13</td>
<td>17</td>
</tr>
<tr>
<td>150 gram</td>
<td>14</td>
<td>14</td>
</tr>
<tr>
<td>175 gram</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>200 gram</td>
<td>19</td>
<td>15</td>
</tr>
</tbody>
</table>

**Figure 6.3 Classifications as a time series with one classifier**
Predictive Health Monitoring for Aircraft Systems using Decision Trees

Figure 6.4  Classifications as a time series with three classifiers

Figure 6.5  Genetic optimization performance
6.3 Condition Prediction

The goal is to predict a typical system health condition function[8]. The function should be followed as close as possible. The experiment and results are shown in[5]. [34] shows how the concept can be used for long horizon forecasts. [34] also evaluates time series models like moving average, exponential smoothing, ARMA and ARIMA.

6.4 Summary

This section showed that the developed concepts work and that it is possible to make a reliable condition monitoring and condition prediction. Condition monitoring was tested on a test system developed by Airbus Operations GmbH. Condition prediction was tested for with a short forecasting horizon and a general system condition function.
Predictive Health Monitoring for Aircraft Systems using Decision Trees
The document showed that it is possible to monitor and predict the condition of a system with the developed concepts. The developed concept combine the use of decision trees and a genetic algorithm with signal analysis and approximation methods to create a system that can learn and adapt to various problems. This does have the advantage that the concept can be easily understood by a human operator and that it can be applied to different monitoring problems. The uses of a decision tree together with the optimization algorithm during the training of the system condition monitoring enables to process to use only the signal analysis methods that give the most useful information to solve a given problem. During the system condition monitoring the decision tree uses the data from the signal processing to classify the signal data and detect the current system condition. The advantages for the forecasting are similar. The decision tree and the optimization are used to select different forecasting methods based on past experience. This does have the advantage that the forecasting method can be switched dynamically during the prediction to enable the process to react to events and to handle non-linearities in the observed data.

Experiments showed that training and using multiple classifiers for the same problem and then take the class which the majority selected improves the accuracy of the classification. For fuzzy decision tree evaluation this means that the similarity value of the other classes needs to be averaged over the decision trees that decided for the same class.

The processes work mostly autonomous and do not require much human interaction. Only for collecting and labeling sample data are humans needed. The algorithms can work alone after the data is setup.
Due to the simplicity of the concepts it is possible to change the algorithms and include new functions. Different decision tree algorithms and time series models can be used.

Future improvements would include improving the preprocessing for the condition monitoring, better labeling of samples for condition prediction and combination of short and long horizon forecasts. The concepts can not replace another maintenance method on the fly. Instead the system can be operated in parallel to the currently used maintenance method. Sensors collect data and they are correlated with maintenance action and system/machine age. After some time the new maintenance concept can replace the old one.
Bibliography


77
Predictive Health Monitoring for Aircraft Systems using Decision Trees


Papers

The articles associated with this thesis have been removed for copyright reasons. For more details about these see:

http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-105843