The University of Nottingham

Department of Civil Engineering

Nottingham Transportation Engineering Centre

Fault Detection and Diagnosis Methods for Engineering Systems

Marius Vileiniskis

Thesis submitted to the University of Nottingham for the degree of Doctor of Philosophy

December 2015

Abstract

The main aim of this thesis is to investigate available techniques and develop a methodology for the fault detection and diagnostics for two engineering systems, namely railway point systems and three-phase separators. The two systems represent two different situations occurring when dealing with fault detection and diagnostics. The first system has only one observable sensor which represents the operation of the system, while for the second system several sensors are available to monitor the operation of the system.

The fault detection of the railway point systems (RPS) was performed on the measured current from the motor of point operating equipment (POE). A threshold based alarm technique is commonly used by railway infrastructure operators. The inability to timely detect the incipient faults of the RPS is a big deficiency of such a technique. The method of One Class Support Vector Machines (OCSVM) together with some elastic metrics has been proposed to overcome this weakness of the threshold based alarm technique.

The OCSVM has been chosen as the fault detection model. OCSVM is a classification algorithm designed to distinguish between two classes when data of one class is underrepresented. Two different approaches of using similarity measures with OCSVM model have been proposed. The first approach was to use the similarity measures as replacements of the OCSVM kernel distance function. The second approach was to use the outputs of the comparison between two consecutive movements of POE, carried out using the similarity measures, as inputs to OCSVM. Their effectiveness has been tested and the first approach proved to perform better.

The standard Euclidean distance used to compare the similarity of two time series was not capable to deal with certain features of in-field RPS data. Elastic similarity measures, such as edit distance with real penalties (ERP) and dynamic time warping (DTW), were chosen to compare the data of POE operations. A combination of Euclidean distance and elastic similarity measures has been proposed in order to take into account the absolute values and shape properties of the two compared time series. Several pre-processing techniques have been considered to get a better estimate of the similarity between two time series, as well as to decrease the computational time taken to obtain the elastic similarity measures.

The proposed methodology has been tested on the in-field RPS data. The results indicated that the fault detection model was able to detect changes in absolute values and/or shape of the time series of measured current. However, not in all cases these changes could be related to a recorded failure of RPS in the database. Furthermore, deficiencies were found in recording faulty and fault-free RPS condition in the database available in this study. Such deficiencies prevented further development of the methodology and practical usage.

The fault detection of three-phase separators (TPS) on chemical process plant was performed given the sensor readings of flow and level transmitters of TPS. A threshold based alarm technique is commonly used by oil and gas infrastructure operators to monitor the operation of TPS. The late detection of faults of the TPS is a big deficiency of such a technique, since it causes the oil and gas processing plants to be shut down. The method of Bayesian Belief Networks (BBN) has been proposed to overcome this weakness of the threshold based alarm technique.

The BBN has been chosen as the fault detection and diagnostics model. BBN is a causal graphical network designed to model causal relationships between variables

in a probabilistic way. An approach to observe sensor readings of TPS in several adjacent time intervals and to update the prior probabilities in the BBN after inserting the sensor readings as evidence was proposed. Generic rules to build the BBN in a modular way for TPS which can be easily adjusted to similar systems were proposed.

A TPS simulation model was built and software with graphical user interface was developed to perform the simulation of selected failure effects on the TPS. Simulation models are favoured for the testing of fault detection and diagnostics techniques, since they can capture main operating conditions of the real systems, are cheap to implement and even hazardous failure modes can be easily tested. The proposed methodology has been tested on the data obtained from a TPS simulation model. The results indicated that the fault detection and diagnostics model was able to detect inconsistencies in sensor readings and link them to corresponding failure modes when single or multiple failures were present in the TPS. However, not in all cases the multiple failures were detected, mainly due to the hidden failures. Further testing of the methodology on in service TPS is needed before using it in practice.

Acknowledgements

First and foremost, I would like to express my gratitude to Dr Rasa Remenyte-Prescott, Dr Dovile Rama and Professor John Andrews for believing in me and giving me the opportunity to pursue a PhD degree. Thank you for your patience and continuous support throughout the years we spent together.

I would also like to say special thanks to Dr Darren Prescott for giving me advice and testing my knowledge as an internal assessor throughout my PhD research. Furthermore I would like to thank the University Of Nottingham for funding my PhD studies and providing the highest level facilities and resources for me to perform the research.

Nottingham Transportation Engineering Centre has been a wonderful place to study and work in. I want to thank the personnel as well as my friends Bryant, Matthew, Claudia, Yang, Jingyu, Diana, Serena, Laura, Raph, Jack, Sara, Tanguy and all other people that have been spending their lunchtime playing cards. I would also like to thank the NUBC and coach Martin Richardson for letting me experience the competitiveness of British university basketball. Lithuanian Society basketball team gave me a great amount of opportunities to celebrate victories and clear my mind from stress. I hope we will meet not only in the basketball court once more.

My warmest thanks go to my wife Šarūnė and our little son Joris. Your continous support and love was one of the key drivers for me not to give up in harsh times. Thank you for being the most amazing person in the world, a wonderful wife and a friend, this thesis would not be possible without you. My gratefulness goes to my father Virginijus, my mother Lilija, my brother Tadas and my sister Eglė for their continuous support and belief. All of you have been my role models and I would never be the person I am without you and I dedicate this thesis to you all.

Nomenclature

AID – Asset information database

BBN – Bayesian Belief Networks

CO – control output

CPT – Conditional probability tables

CVi – ith control valve

δ_E – Euclidean distance using non-scaled data

δ_{ERP} – Edit Distance with Real Penalties using non-scaled data

δ_{DTW} – Dynamic Time Warping using non-scaled data

δ_{EE} – Edit Distance with Real Penalties combined with Euclidean distance using non-scaled data

δ_{DE} – Dynamic Time Warping combined with Euclidean distance using non-scaled data

δ_E^R – Euclidean distance using rescaled data

δ_{ERP}^R – Edit Distance with Real Penalties using rescaled data

δ_{DTW}^R – Dynamic Time Warping using rescaled data

δ_{EE}^R – Edit Distance with Real Penalties combined with Euclidean distance using rescaled data

δ_{DE}^R – Dynamic Time Warping combined with Euclidean distance using rescaled data

DTW – Dynamic Time Warping

ERP – Edit Distance with Real Penalties

FLD – Fault logging database

FC – failed closed

FH – failed high

FL – failed low

FMEA – Failure Mode and Effects Analysis

FO – failed opened

FS – failed stuck

FTi – ith flow rate transmitter

GUI – Graphical user interface

K-NN – K-Nearest Neighbours

LCi – ith level controller

LTi – ith level transmitter

MLD – Movement logging database

NN – Neural Network

OCSVM – One Class Support Vector Machines

OOBN – Object Oriented Bayesian Belief Network

P_G – Classification accuracy of fault-free movements

P_F – Classification accuracy of faulty movements

PCA – Principal Component Analysis

PI – proportional integral

POE – Point operating equipment

RPS – Railway point systems

SVM – Support Vector Machines

TPS – Three phase separators

List of contents

1 Introduction

The research presented in this thesis considers two engineering systems: railway point systems and chemical process plant three-phase separators. The fault detection and diagnostics techniques were analysed in order to propose a methodology how fault detection and diagnostics could be performed for the considered systems. The two systems represented two different situations, which might occur when considering fault detection and diagnostics.

The railway point systems have only one variable which is observed to determine the system condition, i.e. the current of motor of point operating equipment. Due to this the fault detection and diagnostics become a very challenging task, since usually the operation of the railway point system is affected by multiple causes and contextual data would be highly beneficial.

On the contrary, the three-phase separators have several sensors to monitor the operation of the system, i.e. flow and level transmitters. This way the readings of the sensors can be cross-checked in order to identify the operating conditions which drift from an expected behaviour.

These differences influenced the choice of a technique to perform fault detection and diagnostics. The two systems and the techniques chosen in the proposed methodology are briefly discussed in the next sections.

1.1 The need for a fault detection system of railway point systems

An increasing demand for railway transportation requires railway systems to be fault tolerant and secure for both passengers and cargo transport. In 2012/2013, Network Rail, Britain's railway infrastructure operator, was responsible for 8.7 million minutes of train delays: almost 7.4 million of delay minutes were recorded for passenger trains, while the remaining delay minutes were assigned for freight transport (Network Rail, 2013). One of the main causes of train delays was failures of point systems with 575,679 minutes of total delays (Network Rail, 2013). If the target for the delays set out by Office of Rail Regulation is not met, fines are imposed to Network Rail. Network Rail was fined £53.1 million after the 4th control period ended and was obliged to make an additional £25 million investment to improve the resilience of the network (The Office of Rail and Road, 2015).

A railway point system (RPS) allows trains to change tracks, by moving a set of rails from one track to another before the train reaches the crossing. There are a number of different point systems, but they all work in a similar way. The rails are moved in one of two directions: normal to reverse or reverse to normal. Once the movement is completed the rails are locked in their position and the train can pass.

Most problems with point systems are associated with the wear or misalignment of their components. Therefore, railway points require regular adjustment to compensate for the wear or to correct any misalignments (Atamuradov et al., 2009). Even though this process requires a lot of time resources and is very costly, it is essential to keep point systems in a good condition. Poor maintenance and missed faults might lead to deadly accidents, e.g. Potters Bar crash on 10th of May 2002. 7 people were killed and 76 injured in the train derailment in Potters Bar, when the points broke down while the train was passing through (Bagwell, 2004). The last carriage of the train crashed into a bridge turning on the side and eventually crashing into the station, while the locomotive and the first two carriages proceeded up the line.

Several loose bolts were found prior to the crash and they were screwed back again, however the inspection after the crash showed that there were more loosened bolts and the maintenance of the point system was performed poorly.

To avoid such disasters and improve the RPS fault detection and maintenance, researchers all over the world are trying to implement advanced algorithms to detect and classify point system faults. Approaches used to assist the engineers to monitor the condition of RPS can be broadly classified into: condition monitoring systems and fault detection and diagnosis systems. Condition monitoring systems are used to monitor the behaviour of observed parameters of RPS, e.g. current or power consumption of railway point operating equipment motor. However, they do not provide any diagnostic capabilities. Condition monitoring systems are increasingly used by infrastructure operators. They often contain an alarm system based on the threshold technique. A threshold for faulty behaviour of observed parameter has to be predefined and it is usually chosen based on expert knowledge. However, the choice of the threshold is not intuitive and requires careful consideration throughout a certain length of time to incorporate the changes of operating conditions of the railway point systems.

A fault detection and diagnosis system is a more advanced version of a condition monitoring system, incorporating 'intelligent' algorithms. These systems are of interest to railway engineers and maintenance personnel because they can be capable of detecting a deteriorating condition prior to a system failure and diagnosing incipient faults. A methodology of how the fault detection of RPS could be approached instead of using alarm threshold technique is proposed in this thesis.

1.1.1 One Class Support Vector Machines for fault detection

The fault detection of RPS can be considered as a classification task, i.e. to classify the movements of railway point system into different classes: one class for good and the other classes for faulty movements. Due to specific features of the data available in this study (to be discussed later), One Class Support Vector Machines (OCSVM) (Scholkopf et al., 2001) are used to classify movements. Their idea is based on the learning of the features of one class in the training phase and then detecting the abnormal data, when the OCSVM is used to classify the new data. Such an approach avoids an issue of using a highly imbalanced proportion of the data available for good and faulty classes.

The OCSVM forms a boundary between a good class and an abnormal class, by picking out the best representatives from the good class in the training phase. These representatives of the good class, which form the boundary of OCSVM, are known as Support Vectors. Only the Support Vectors and their respective coefficients are then used to decide whether the new data should be classified as good or as abnormal. Such an approach, further described in section 2.3.5, allows using only a fraction of training data for classification of new data, which is advantageous for fast fault detection.

1.1.2 Similarity measures of time series

One of the key points in this study, when using the classification methodology, is the definition of the similarity of data of point movements, obtained from infield railway point systems. The main measurement that is available for the analysis is the measurement of current used by a motor moving a set of points. This data is

represented by a time series, of current measurements recorded over a single movement of points. A similarity of two time series is usually represented by the Euclidean distance between these time series. However, the Euclidean distance is only defined for the time series which are of equal length. Due to the nature of a railway point system, any movement of the switch rails might take different time to complete and thus it results in time series of different length. The Euclidean distance cannot be used on the raw data in this case and some data pre-processing has to be considered, which is presented in section 2.4.3.1.

Alternatively, different similarity measures that are defined for the time series of different length can be used. These include Dynamic Time Warping (DTW) (Sakoe and Chiba, 1978), Edit distance with Real Penalties (ERP) (Chen and Ng, 2004), Time Warp Edit Distance (TWED) (Marteau, 2009) and others. Their idea is based on allowing elastic matching by comparing data points which are recorded at different time points, deleting an observation or repeating the same observation a number of times in the time series. Each of these operations is penalised in some way, specific to each similarity measure. The majority of commonly used similarity measures of time series are defined in detail in section 2.3.6, presenting the way they allow elastic matching and how the operations of elastic matching are penalised. A combination of elastic similarity measures (ERP and DTW) and Euclidean distance was proposed in this thesis. The resulting similarity measures are defined in sections 2.3.6.5 and 2.3.6.6.

1.2 The need for a fault detection system of three-phase separators

According to the U.S. Energy Information Administration approximately 90 million barrels of oil products were produced every day around the world in 2013 (U.S. Energy Information Administration, 2013). An increasing demand for oil products requires the oil and gas processing facilities to work at maximum efficiency and every second spent without producing oil results in economic losses. The biggest losses are experienced, when the offshore oil and gas plants need to be shut down unexpectedly due to a failure of its equipment. Three-phase separators are one of the key components of offshore processing facilities. A failure in the three-phase separator can cause the whole oil processing plant to be stopped. Thus, a timely detection of failing components in the three-phase separators is necessary.

Faults in three-phase separators are commonly detected by using either thresholds of the process variables (e.g. oil level, water level and etc.) or precise mathematical models simulating the operation of the three-phase separator and then comparing its outputs with the readings of the actual separator. The first approach usually detects failures when their effect is already critical and the prevention of the shutting down of the separator is unavoidable. The second approach needs a very good understanding of the process conditions and usually requires extensive changes if operating conditions change.

A novel fault detection and diagnostic methodology for three-phase separators is proposed in this study. It can give an early warning of a failure in the system and has an ability to be easily adapted for the specific system. The methodology was built using the Bayesian Belief Network (BBN) technique. Such an approach was chosen due to several reasons, including graphical representation of the modelled system, inclusion of expert knowledge about failure modes of the system, ability to model uncertainties in a probabilistic way, ability to build the model in a structured and modular way and update the prior knowledge about occurrence of certain failure modes without altering the structure of the model.

1.2.1 Bayesian Belief Networks for fault detection

BBN is a common technique used for the fault detection of industrial systems. BBNs are directed acyclic graphs consisting of nodes (they can represent process variables, states of components of the system) and arcs between the nodes, which describe causal relationships between the nodes. Causal relationships between nodes in a BBN model are quantitatively determined by conditional probability tables (CPT), which represent the likelihood of a node being in a particular state given the states of other nodes and allow uncertainty to be included in the system model. The latter feature is highly advantageous when modelling systems (usually combined from multiple subsystems) where a precise mathematical model of the system cannot be developed.

Another important feature of the BBN technique is that models can be built in a structured way, for example, using the same logic as that used to build Fault trees, which are commonly used in industrial applications due to their intuitive representation of possible failure scenarios. A similar approach was taken in this thesis to build the BBN in a structured way, however the fault trees were only used as an initial state of the research to develop generic rules for building the BBN for three-phase separator. The detection and diagnostics of three-phase separator faults was performed using the BBN by monitoring the probabilities of certain states of the nodes, which represented states of the components of three-phase separator. The information from various transmitters of the system was inputted into the BBN model resulting in changes in the probabilities of BBN nodes. These changes were the decisive factors for determining of a certain failure occurrence. More details on fault detection and diagnostics using the BBN can be found in section 3.4.3.

1.3 Research objectives

The main aim of this research is to create fault detection and diagnostics tools for two engineering systems given readings from sensors, which inform about the operation of considered systems. The objectives that were set for both systems were similar:

- Propose a methodology for the fault detection and diagnostics of the considered system;
- Implement the proposed methodology in software;
- Validate the proposed methodology on data representing the operation of considered system.

The research objectives that were specifically chosen for each considered system are presented in the next subsections.

1.3.1 Research objectives for the fault detection of railway point systems

The main aim of this research is to create a fault detection tool for the railway point systems, based on measurements of current, which could identify faults prior to their occurrence. The following objectives have been fulfilled:

- Propose a methodology for the fault detection of railway point systems;
- Implement the proposed methodology in software;
- Validate the proposed methodology on the data obtained from the infield railway point systems.

1.3.1.1 Propose a methodology of how the fault detection of railway point systems could be approached

The main objective of this research is to propose a new methodology for fault detection of railway point systems that could be used with the data of infield railway point systems. A comprehensive literature review has been performed to identify the methods that have already been used for the fault detection and prediction of railway point systems. From the literature review, a gap in research was identified, when the fault detection of railway point systems was considered:

- Majority of the methods were validated on small sample sizes of data obtained from laboratory RPS, which did not represent the operating conditions (e.g. severe weather conditions, presence of big forces when the train is passing through RPS etc.) of the infield RPS well enough.

- Similar amount of data representing failures and good movements was needed to train and test the proposed methods. However, the available amount of data representing failures of infield RPS is usually significantly smaller than the amount of data representing failures obtained from the laboratory RPS.

- The methodologies proposed were only able to detect specific faults that were present in historical data.

- Majority of the methods were trained and tested to detect the failure when one of the observed parameter had abruptly changed (e.g. current drawn by

POE, power used by POE, duration of operation etc.). This makes the detection of a deteriorating condition impossible.

Based on these findings, requirements for a new fault detection technique of infield railway point systems have been formed and they will be presented in section 2.2.5. A new methodology has been proposed for the fault detection of railway point systems using measurements of current.

1.3.1.2 Implement the proposed methodology in software

The proposed methodology was implemented in software, so that it could be demonstrated. MATLAB was chosen as a graphical interface and data input-output software, while the main calculations were done by calling C++ LIBSVM package from within the MATLAB platform. Such an approach allowed the user to take advantage of the running speed of C++, while the MATLAB was used for the visualisation of results. Special functions were written to input data from MATLAB into C++ and convert the results from C++ to MATLAB. Results showed that the C++ speed advantage over MATLAB was very significant.

1.3.1.3 Validation of the approach using data obtained from infield railway point systems

The proposed methodology for the fault detection of railway point systems was validated on data obtained from infield railway point systems. The majority of the researchers so far validated their approaches on data obtained from laboratory point systems. However, the operating conditions of infield railway point systems can be different from the ones that are provided in the laboratory test field, therefore, some important features of real data can be missed out and not incorporated in the models, if they are based on the data obtained from laboratory railway point systems only. Moreover, the features of data obtained from infield railway point systems also introduce difficulties for the timely and accurate detection of faults. Thus the results, presented in this thesis, illustrate the performance of the proposed methodology if it was to be applied on a real RPS.

1.3.2 Research objectives for the fault detection and diagnostics of three-phase separators

The main aim of this research is to create a fault detection and diagnostic tool for the three-phase separator, which could identify faults of components before an alarm is raised in the system or an undesirable event develops, based on the readings of flow and level transmitters of the three-phase separator. The following objectives have been fulfilled:

- Propose a methodology for the fault detection and diagnostics of three-phase separators based on BBN technique;
- Develop the software to implement the methodology;
- Develop a simulation model of the three-phase separator and its software;
- Validate the proposed methodology on the data obtained from the simulation model of the three-phase separator.

1.3.2.1 Propose a methodology for the fault detection of three-phase separators

In this study, a methodology for the fault detection of three-phase separators based on the method of BBN was proposed. Generic rules for building BBNs for individual sections and for the system as a whole were formulated, allowing the development of the BBN in a structured way. The whole system BBN model, known as Object Oriented Bayesian Belief Network (OOBN), was then used to analyse sensor readings of the three-phase separator, when components of the separator were free from faults and when faults were present. The fault detection and diagnostic stages were performed simultaneously. The posterior probabilities of the states of nodes in the OOBN that represented the condition of the components of the three-phase separator were updated as new system sensor readings were made available to the OOBN. An increase in the posterior probabilities of the states of certain nodes that represented the fault of a certain component was considered as a sign of a fault of that component.

1.3.2.2 Develop software to implement the methodology

The OOBN model was implemented using the commercial software HUGIN. The OOBN and individual BBNs were built using the graphical user interface of HUGIN and were then used for fault detection and diagnostics by exploiting the HUGIN Application Program Interface (API) for C++. The use of the API allowed a custom software to be created for the automated data input (sensor readings of a three-phase separator) into the developed OOBN model. It also allowed producing an automated summary of results with plots of posterior probabilities for each failure mode over time. The analysis of the OOBN model was very fast, thus the proposed methodology has the potential of being implemented as an online monitoring and fault detection and diagnostics tool for three-phase separators.

1.3.2.3 Develop a simulation model of the three-phase separator and its software

A model of the three-phase separator for simulating the operation of the system was built using C++. A graphical user interface (GUI) was developed with wxWidgets for C++ to allow the user to perform numerous tasks such as to interactively insert selected failures of components at specific times, to change the type of inflow as well as physical parameters of the three-phase separator, to plot results from the simulation and export readings of various transmitters for the fault detection and diagnostics stage. The developed GUI and the simulation model are presented in section 3.3.

1.3.2.4 Validation of the approach using data obtained from a three-phase separator simulation model

The validation of the proposed method is a necessary step in order to evaluate the performance of the method proposed for fault detection and diagnostics when the three-phase separator operates under different conditions. The validation also allows the strength and weaknesses of the proposed approach to be identified.

The proposed methodology for fault detection and diagnostics was validated with a number of scenarios by inserting single and multiple failures at different time points

and using different inflow options in the simulation model of the three-phase separator. The sensor readings obtained from the simulation model were used as inputs to the BBN. Such a validation method is commonly used for the initial testing of fault detection and diagnostic techniques of industrial systems, due to its low cost, the ease of repetitive testing for a number of distinct failure scenarios and the possibility to test the ability to diagnose critical failure modes, which could lead to hazardous events.

The rest of the thesis is structured in the following way:

- Study on fault detection of railway point systems is presented in chapter 2.
    - An overview of the railway point system, its components, failure modes and features of the data available in this study is presented in section 2.1.
    - A literature review of the state of the art methods for fault detection and prediction of RPS is given together with the common industrial solutions for RPS fault detection in section 2.2.
    - Possible approaches for the fault detection of RPS and similarity measures to compare the measurements of current of POE are presented in section 2.3.
    - The proposed methodology with OCSVM as fault detection model, giving the details of data input, pre-processing techniques and chosen training and testing techniques of OCSVM, is presented in section 2.4.
    - Results with the proposed methodology applied on the data of infield RPS, highlighting the effects of different pre-processing techniques and similarity measures used, are presented in section 2.5.
    - Conclusions and future recommendations for the RPS study performed in this thesis are given in sections 2.6 and 2.7 respectively.
- Study on fault detection and diagnostics of three-phase separators is presented in chapter 3.
    - An overview of the three-phase separator, its components, undesirable events, failure modes and automatic control is presented in section 3.1.
    - A literature review of the state of the art methods for fault detection and diagnostics of TPS is given in section 3.2.
    - Three-phase separator simulation model and the developed software are presented in section 3.2.2.5.
    - The proposed methodology with BBN as fault detection and diagnostics model for TPS is presented in section 3.4.
    - Results with the proposed methodology applied on the data from TPS simulation model with single and multiple failures are presented in section 3.5.
    - Conclusions and future recommendations for the TPS study performed in this thesis are given in sections 3.6 and 3.7 respectively.
- Final remarks are given in chapter 4, followed by references and appendices.

2 Study on fault detection for railway point systems

2.1 Railway point systems

2.1.1 Introduction

In the next subsections an introduction to railway point systems is given. All main components of the system are presented and the operation of point system is described. Then possible operating problems are introduced by listing the component and system failure modes that occur commonly. Afterwards, maintenance and fault rectification intervals and regimes are introduced. The condition of a railway point system can be observed by monitoring several parameters of the system, which are introduced in this section. The behaviour of these parameters when no failure is present in the system is described. Then the expected change of these parameters behaviour, when the failure occurs in the system, is given. Finally, data available for this study is presented. The data available from several databases is combined to obtain a dataset required for the analysis of fault detection of railway point systems. Main features of the data of infield railway point systems are presented, highlighting the ones that differ from the data obtained from laboratory RPS.

2.1.2 Railway point system operation and components

A railway point system (in literature also referred to as switches and crossings (S&C)) is combined of electrical and mechanical installations enabling railway trains to be guided from one track to another. An example of a railway point system can be seen in Figure 2.1. Main components and their roles in the operation of the railway point system are discussed next.

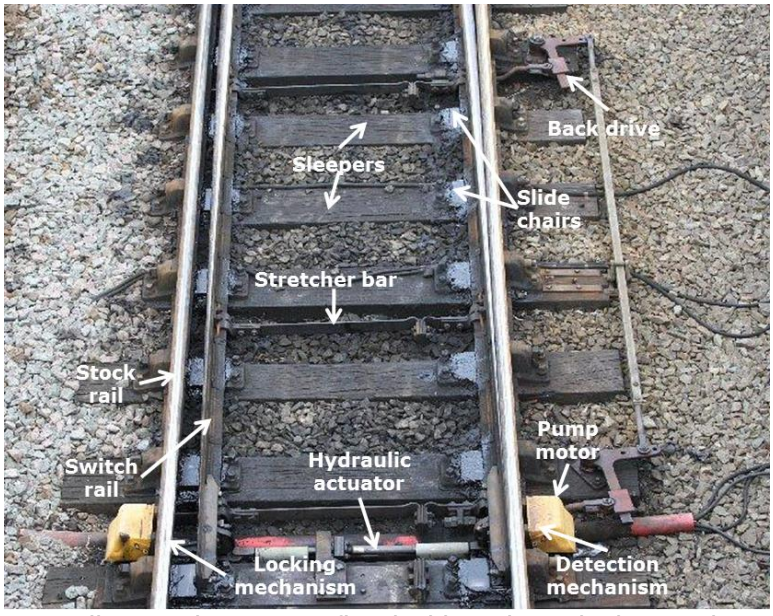


Figure 2.1. Railway point system fitted with a clamp lock as point operating equipment (Skipsey, 2010)

Railway sleepers can be wooden (as in Figure 2.1) or concrete rectangular shaped objects that give the support for the rails, which are fastened to the sleepers. Stock rails are the rails which are not moveable in a railway point system. A pair of linked

movable rails lying between the stock rails are called switch rails. The switch rails are maintained at a correct distance apart by stretcher bars. The switch rails are usually laid on additional metal structure, called slide chairs. To reduce the friction between slide chairs and switch rails, the slide chairs are greased regularly. An additional component that helps to move the switch rails is known as a back drive. It moves the switch rails with the help of an additional stretcher bar and ensures that the switch rails are moved evenly. The number of back drives and stretcher bars can vary between installations depending on the length, curvature and speed limit of the railway point system. The drive of the switch rails is provided by a point operating equipment (POE). Various types of POE can be used within a railway point system. In this case, the railway point system is operated by a clamp lock (hydraulic POE). Its drive is provided by pumping oil through a hydraulic circuit.

Other commonly used POE in UK railways is GEC HW electric point machine and Westinghouse M63 electric point machine. Depending on the type of POE, some components of RPS may differ. For example a drive rod together with a stretcher bar is used to move the switch rails, when HW is used as POE, while the clamp lock uses a hydraulic actuator and a stretcher bar to move the switch rails. In this study, railway point systems with the clamp locks are considered, but the same fault detection methodology can be applied to point systems with other types of POE. Since the measurements of current depend on the type of POE, the same idea of comparing the shapes of the measurements of current can be used.

The clamp lock consists of a power pack, two pump motors and a hydraulic actuator. Two movement directions are identified: normal to reverse or reverse to normal. For each direction a different pump motor is used. If the switch rails were to be moved from the position that is pictured in Figure 2.1, such a movement would be called reverse to normal. The operation of RPS can be described as follows:

1. A signal is received by the RPS to change the direction of track from normal to reverse or reverse to normal. The oil pump motor starts up (motor start-up phase in Figure 2.2).
2. The switch rails are unlocked (unlocking phase in Figure 2.2).
3. The corresponding pump motor pumps oil to the hydraulic actuator, which together with the back drive moves the switch rails (switch rail movement phase in Figure 2.2).
4. When the switch rails move to the desired position, they are locked and a signal is sent to the signaller that the railway point system has successfully completed the operation (locking phase in Figure 2.2). Due to a slight delay in detecting the locking, the motor keeps running after the points have completed their movement. This is common to clamp lock and thus a ‘shark fin’ at the end of the operation is observed, as shown in Figure 2.2.

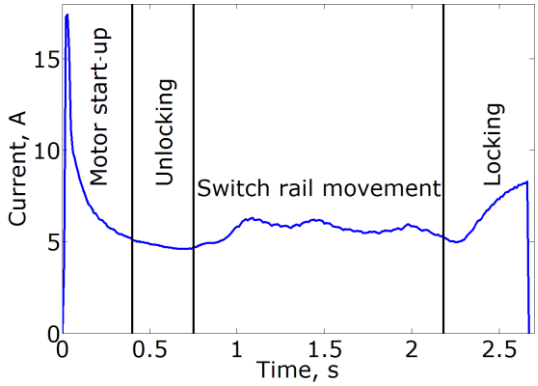


Figure 2.2. Operating phases of the clamp lock in a typical electric current trend

Various failure modes might occur during this operation and they are presented in the following sections.

2.1.3 Failure modes of railway point system

The successful operation of a railway point system mostly depends on the reliable performance of its components. However, some externals factors might disrupt the operation even if all of the components of the railway point system are working. Both component failures and external causes that result in a failure of a railway point system are described in the next sections.

2.1.3.1 Component failure modes

Stretcher bar. A stretcher bar can fail in two ways. It can become loose from the switch rails or it might break due to corrosion or overstress. In the case of a failure of the stretcher bar, the switch rails are moved unevenly. This should result in some abnormal behaviour during the switch rails movement phase. The switch rails might even be moved at one end of the point system, but not at the other, depending on the number of stretcher bars used. A hazardous situation for railway users would occur if the railway point system was used with this failure present.

Slide chairs. There are two failure modes of slide chairs. One of them is dry slide chairs, which might occur due to heavy rain or inappropriate maintenance frequency. The grease on the slide chairs might be washed off after a heavy rain and thus increase the friction between the switch rails and slide chairs. The other failure mode is the contamination of slide chairs. The slide chairs might get contaminated by sand, crushed ballast or fallen leaves. Both failure modes, dry or contaminated slide chairs, increase the friction between the slide chairs and switch rails. Thus, the railway POE has to put more effort in, in order to move the switch rails.

Pump motor. A failure of a pump motor would result in the lack of oil delivery to the hydraulic actuator. In such case, after the unlocking phase, no movement of switch rails would be possible and the pump motor of clamp lock should time out after running longer than usual.

Hydraulic actuator. A failure of a hydraulic actuator, as in the case of a failure of the pump motor, would also result in no movement of switch rails after the unlocking phase. The pump motor of clamp lock should also time out.

Back drive. The failures attributed to the back drive are usually due to misalignment. The back drive might be under driven or over driven. The under driving of the back drive means that the switch rails are not moved to their end position properly, leaving a bigger than usual gap between the stock and switch rails. The overdrive of the back drive means that the back drive forces to move the switch rails further when they already are at their end position. The overdrive might cause the stretcher bar to crack due to overstress.

Locking mechanism. The locking mechanism might fail in two ways: fail to unlock the switch rails and fail to lock the switch rails. In the first case, the switch rails cannot be moved from their position and thus the railway point system is inoperable. In the second case, a more hazardous situation occurs. If the train passes the railway point system when the switch rails are not properly locked, the switch rails might move while some of the train carriages are still passing through. This could result in a

catastrophic event of train derailment. The degraded condition of the locking mechanism might be detected by an increase of current used in the locking phase.

Detection mechanism. The failure of the detection mechanism would result in an undetermined or an incorrectly determined position of the switch rails. If the locked position of switch rail is not detected, the point system cannot be used, because it is unclear if the operation of switch rail movement was performed successfully. A more hazardous situation might occur if the position of the switch rails is falsely identified as locked. As in the case of a failure of locking mechanism, this could result in a train derailment.

Failures of sleepers, switch rail and stock rail are not considered in this section. The reasoning behind this is that such failures would not be observed from the behaviour of the usually monitored parameters (presented in section 2.1.5) of a railway point system.

The following component failure modes of UK railway point systems were identified as the most common ones (INNOTRACK, 2006):

- Failure of detection mechanism.
- Dry or contaminated slide chairs.
- Failure of back drive.
- Failure of locking mechanism.

However, situations described as “no fault found”, were more frequent than any failure mode listed in a study conducted by INNOTRACK (INNOTRACK, 2006). Very often, when maintenance engineers are called to rectify faults of point systems, they cannot find any particular problem as the point system operates normally at that time. This might be due to several reasons. The first reason is that the fault occurs spuriously and the cause of such fault might depend on seasonal conditions (e.g. low temperature, heavy rain, flooding, etc.) which change in a time span between the fault detection and the arrival of maintenance team on site. The second reason is that the fault detection system raised a false alarm. Such uncertainty in the actual condition of RPS makes it difficult to differentiate between the data, representing working and failed state of the RPS.

2.1.3.2 External causes

Obstruction. One of the frequent external causes that alter the operation of a railway point system is an obstruction of the gap between stock rails and switch rails. Pieces of ballast, empty cans, bottles or any type of object that might get into the railway point system from the surrounding area might cause problems to the successful operation of the system. The obstruction in the system forces the POE to use more force when moving the switch rails. If the obstruction is not crushable, the POE might fail to complete the switch rails movement.

Icing. Icing can alter the operation of locking and switch rail movement phases of the railway point system. If the switch rails are frozen, the POE has to use more force to start moving the rails. Moreover, if the locking mechanism is frozen, the unlocking of the switch rails could fail, resulting in the failure to operate the point system. Icing is temperature dependant external cause and is more likely to occur in the regions, where the weather conditions are more severe during the winter.

2.1.4 Maintenance and rectification of faults

Two types of maintenance can be performed on a railway point system. The first type is preventive maintenance, which is performed at certain time intervals, disregarding the condition and performance of the point system. The second type is corrective maintenance, carried out by the engineers, only when a faulty behaviour of the RPS has been identified: for example, points fail to lock in the desired position. Using a strategy of corrective maintenance only, unpredictable delays on the network might occur, or in the worst case, a disastrous event might occur.

Preventive maintenance is a safer, but usually more expensive way, of maintaining the RPS. This type of maintenance has a regular schedule, which is determined for each point system. The schedule is determined based on the risk it is exposed to, for example, the number of trains passing per day or the number of switch rail movements per day. The maintenance could be performed from as frequently as 4 week intervals for high risk point systems, and as rarely as 52 week intervals for low risk point systems. A visual inspection is carried out during the maintenance. Any faults found are rectified immediately if possible. If the fault cause has not been found or the fault cannot be rectified straight away, the line might be closed or a temporary line speed restriction might be introduced, until the failure is rectified. As part of the routine maintenance, the cleaning and greasing of slide chairs for a smoother movement of switch rails is also performed. In this study, only the information about the corrective maintenance was available and no records of preventive maintenance were obtained. Preventive maintenance schedule can also be based on the condition of the RPS. In the next section, a way to monitor the condition of railway point system is presented.

2.1.5 Measurements for monitoring the condition of railway point systems

Several parameters on the railway point systems can be observed in order to monitor the condition of the system. Measurements of current and force used by POE and the displacement of switch rails are the most commonly observed parameters. These parameters were found to be most affected by incipient faults (INNOTRACK, 2009). Among them, the effect on displacement parameter was of the smallest magnitude. All measurements are usually taken via non-intrusive way, i.e. the introduction of sensors for current, force and displacement measurement does not affect the behaviour of the system in any way. The data used from infield railway point systems was obtained in this study as measurements of current. The current is measured using a current transducer, which is installed on the current supply to the pump motor of a clamp lock. The current drawn during each operation of the RPS is recorded in the database, identifying the start and the end of the operation together with intermediate measurements throughout the operation.

In the next section the expected behaviour of faulty systems is presented. Different fault types will be considered to see whether a distinct failure type can be identified from the changes in the measurements of current.

2.1.5.1 Expected behaviour of faulty railway point system

Marquez et al. looked at the failure modes of a specific RPS with type M63 point machine (Márquez et al., 2007a), however some failure modes are common to all types of POE. The authors tried to determine current profiles of the POE during point movement when different types of faults were present. Three failure modes common

to all railway point systems were mentioned, namely obstruction in the point system, dry slide chairs and contaminated slide chairs. The expected profiles of the current measurements for these failure modes were described as follows:

Obstruction. The current profile should be normal until the switch hits the obstruction. The motor current should increase while attempting to overcome the obstruction and if the obstruction cannot be overcome, the motor will time out after certain time. The points will not lock, so the increase of current will be observed until the motor times out. If the obstruction can be forced out of the way, the increase of the current drawn should be observed until the point, when the obstruction is overcome. Afterwards, the points will lock normally.

Dry slide chairs. When the slide chairs are too dry the motor current should increase. The dry slide chairs are expected to increase the friction between the switch rails and slide chairs.

Contaminated slide chairs. When there is sand or leaves on slide chairs, the friction between slide chairs and the switch rails is expected to increase, similarly as in the case of dry slide chairs. This increase of friction should be seen in the increase of the current drawn by the point machine.

From the communication with railway infrastructure engineers, profiles of current measurements, which represented different faulty conditions and failure modes (obstruction, dry slide chairs, back drive out of adjustment and tight lock), were obtained and are presented next. All the numerical information has been removed due to confidentiality.

Obstruction. The first failure mode, described by railway infrastructure engineers, was an obstruction, forcing the POE to time out. As presented previously, existence of obstruction should be indicated by an excess of current used, when the obstruction is hit. Then two scenarios can occur: the obstruction is not forced out of the way and the motor of POE times out after running for a long time or the obstruction is crushed and the railway point system completes the operation. The latter scenario can be observed from Figure 2.3 (a). Here the movement with the fault present is represented with blue colour and a fault-free movement with brown colour.

Dry slide chairs. A similar situation to that of an obstruction can be observed for dry slide chairs as shown in Figure 2.3 (b). However, this faulty condition leads to the successful completion of the whole operation with a slight delay and excess current required. The blue line represents the measurements of current of faulty movement and the brown line represents the fault-free movement. The dry slide chairs force the POE to put in more effort (draw more current) to move the switch rails, as can be seen in Figure 2.3 (movement in blue is above the movement in brown).

Back drive out of adjustment. The current profile obtained when the back drive is out of adjustment is pictured in Figure 2.3 (c). As previously, blue line represents the faulty condition and brown line represents a good condition of the RPS. As it can be observed from the figure, the motor of the POE had to use more effort, as in the case of the dry slide chairs failure mode. This time, it is due to the fact that the POE had to work harder to compensate for the poorly adjusted back drive. The excess of energy used can be seen throughout the whole operation of the railway point system.

Tight lock. The last faulty condition presented, is known as a tight lock. This faulty condition represents a deteriorated state of the RPS, which would lead to the failure of the locking mechanism if no preventive actions were carried out. The profile of

measurements of current with this faulty condition is also given in Figure 2.3 (d). It can be seen from the figure, that there is an increase in current at the end of the operation of RPS, i.e. locking phase. This indicates that there are some problems in the locking operation of the railway point system.

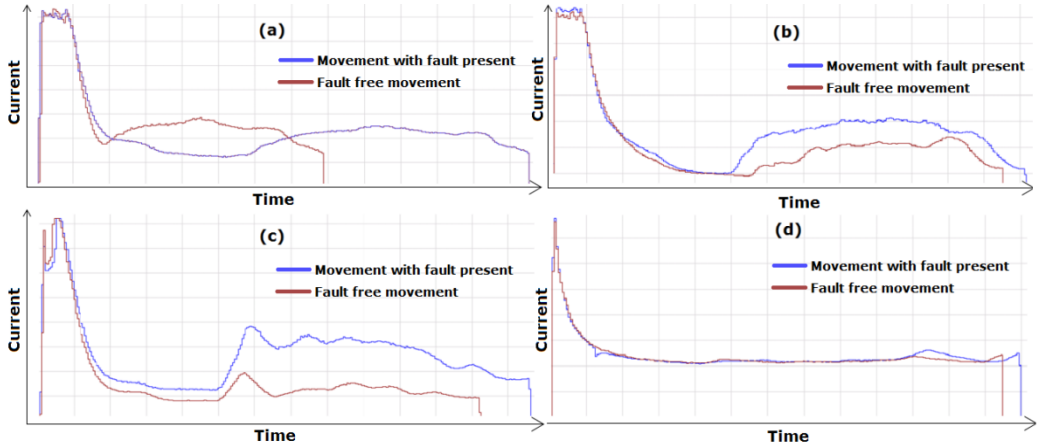


Figure 2.3. Expected profiles of current, as indicated by railway infrastructure engineers, when there is (a) an obstruction in point system, (b) slide chairs are dry or contaminated, (c) the back drive is out of adjustment and (d) there is a tight lock

In general, for the faulty conditions and failure modes, considered from the communication with railway infrastructure engineers and observed in the data available, the excess of current usage was the typical symptom. The effects of failure modes, described by Marquez et al. (Márquez et al., 2007a) were the same as the ones described by the railway infrastructure engineers. The failure modes that were observed in the databases used in this study, will be graphically analysed in the next section.

2.1.5.2 Summary

From the graphical analysis of the expected trends of faulty conditions, one can identify that there is no significant distinction between certain fault types. For example, if dry slide chairs and back drive out of adjustment failure modes are considered (see Figure 2.3), both of them have an increase of current used. The same can be said about the obstruction failure mode. Even though, in the latter case sometimes it results in the timing out of POE. Due to existing similarities among current trends of different failure modes the research performed in this study focused on the detection of faults by distinguishing only between good and abnormal movements. In the next subsections, the databases that were used in this analysis and the data they contained are presented in more details.

2.1.6 In-field data and its main features

2.1.6.1 Databases used

In-field data used in this study came from three databases: movement logging database (MLD), fault logging database (FLD) and asset information database (AID). Data logged throughout a one year period was analysed.

Each movement data extracted from the MLD consists of current measurements (in amperes) over time (in seconds) when the switch rails were moved, the date and time when the movement was started and when it was finished, the POE ID and the direction of the movement (normal to reverse (N->R) or reverse to normal directions (R->N)). Together with the movement data, the corresponding alarms were also extracted from the MLD. The alarms are raised in the MLD if certain thresholds for the observed parameters during the movement of points are reached to indicate possible faults. Two levels of alarms are used: low level alarm and high level alarm. Low level alarms are of lower priority which might indicate a deteriorating condition of the RPS but usually a big percentage of them are false alarms, while high level alarms are usually warnings about a faulty condition that already causes problems to the operation of RPS. The alarms are based on the motor running duration, the maximum and the average values of the measurements of current. More details about the threshold based alarm technique will be presented in section 2.2.4.2.

The FLD consists of information about system faults. Specifically, fault description, repairs or adjustments made to the RPS, the cause of the failure, the time when the failure occurred and the time of repair. The AID is used to identify each RPS and link it to its faults. The type of POE of each RPS is extracted from the AID, since current trends depend on the type of POE.

2.1.6.2 Data cleaning and preparation for analysis

Current data stored in the MLD is the current from POE motor sampled with a 0.01 second sampling rate during the movement of switch rails. In some cases the data extracted from the MLD can be partially incomplete, i.e. contain some missing values, and thus some data filtering and corrective actions are performed before the analysis. Some movements had a NULL value or one value from the whole movement missing (the time difference between two consecutive points in the time series of measurements of current was twice greater than the sampling rate) but the remaining values were present. In these cases a linear interpolation is used to estimate the missing values of current time series:

$$x_{miss} = (t_{miss} - t_{miss-1})\ \frac{x_{miss+1} - x_{miss-1}}{t_{miss+1} - t_{miss-1}} + x_{miss-1}, \qquad \textbf{2.1}$$

where x_{miss} is the missing value of current time series, x_{miss+1} is the value of current time series after the missing value, x_{miss-1} is the value of current before the missing value in time series, t_{miss} is the time when current measurement is missing, t_{miss-1} is the time before the current measurement is missing, t_{miss+1} is the time after the current measurement is missing.

2.1.6.3 Features of the in-field data

A number of specific features have been observed in the in-field data of railway point systems. They have been of great importance in choosing a suitable method for the analysis. Since the data comes from the in-field point systems, the data features are different from the ones that come from the laboratory-based equipment, which was used to develop most of the approaches so far, which will be discussed in section 2.2. The features of the data obtained from the in-field point systems are discussed next, including the diversity of operational behaviour reflected in current trends within a group of same type POE (section 2.1.6.3.1), the variable duration of the RPS

operations (section 2.1.6.3.2) and the changing behaviour of fault-free RPS (section 2.1.6.3.3).

2.1.6.3.1 Diversity of operational behaviour within a group of the same type POE

The idea of grouping RPS with the same type of POE for fault diagnostics is of great interest. That way, large data samples for each type of fault could be obtained allowing the standard multi-class classifiers such as Neural Networks, k-Nearest Neighbours, Naïve Bayes classifiers or Support Vector Machines to be used for the analysis. However, it has been observed that each POE seems to have its own specific current trend and thus POE of the same type cannot be grouped for the analysis. This feature can be simply observed just by looking at some POE current measurements plotted against time. For example, 4 point systems with Clamp locks and the same turnout type have been selected extracting 5 movements in each direction for each individual RPS over one week period of operation.

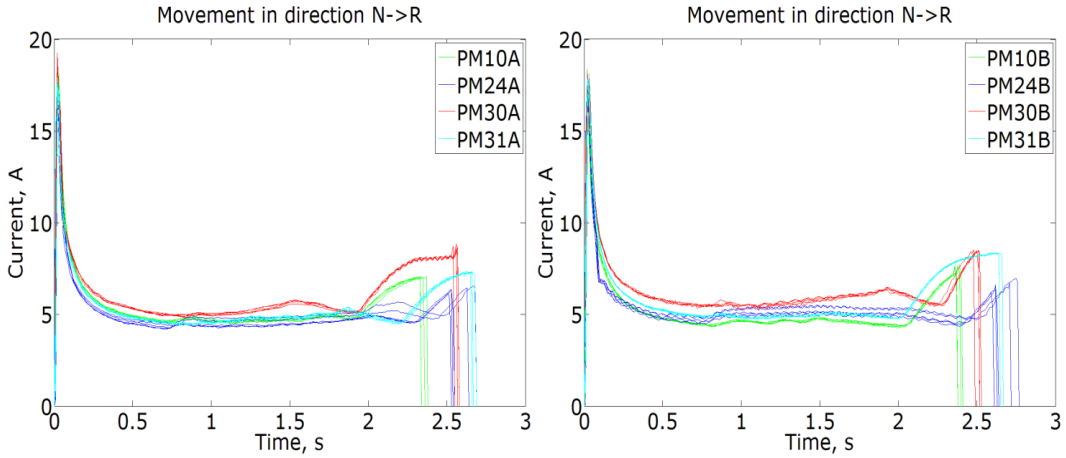


Figure 2.4. Measurements of current of RPS with clamp locks

From Figure 2.4 one can observe that not only the shapes of measurements of current curves are different, i.e. different phases of a movement start at different times, the 'shark fins' at the end of the movement are of different magnitude and also the durations of the movements differ. Larger differences in the shape of current curves can be seen in Figure 2.5, where the first 50 milliseconds (where the shapes are very similar) are ignored.

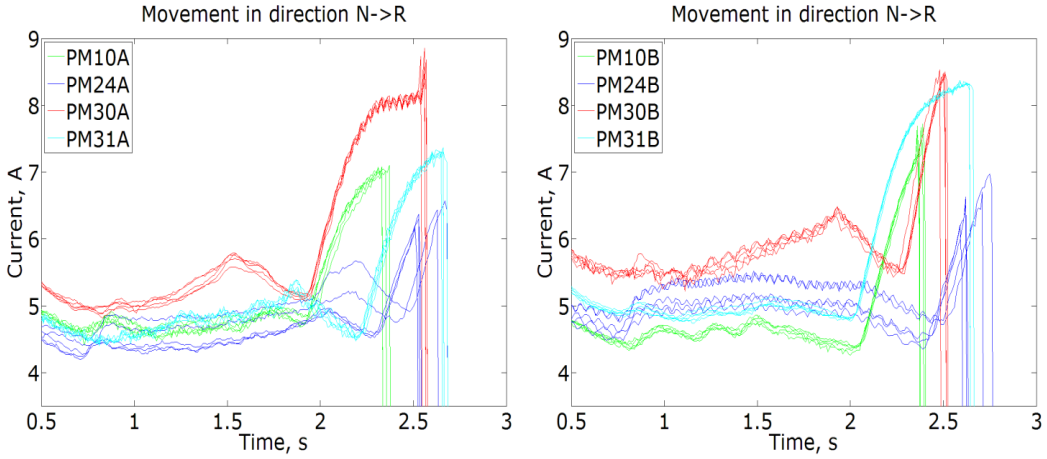


Figure 2.5. Measurements of current of RPS with clamp lock (zoomed in)

Even for two POEs that are used on a single RPS, the shape of the current trend differs substantially, as it can be seen in Figure 2.6. Two POEs on a single RPS is a common thing, when the length of the switch rails is too big to be operated by one POE.

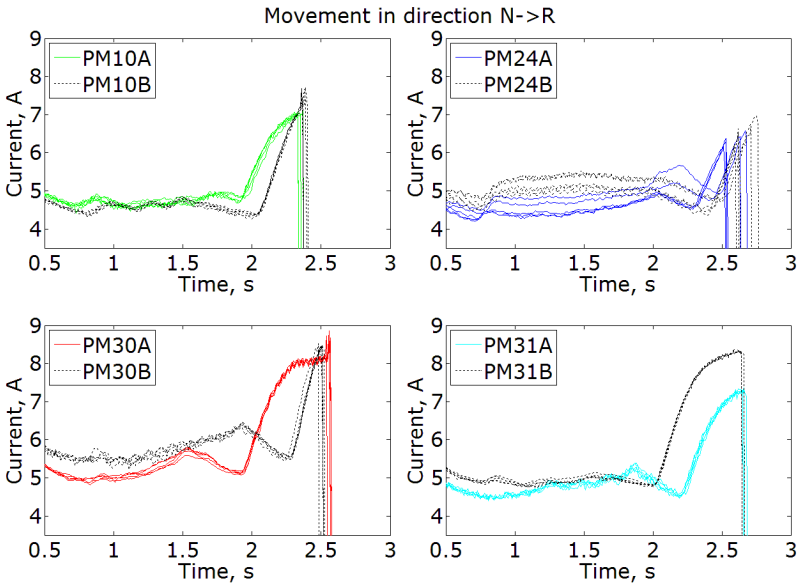


Figure 2.6. Measurements of current of paired clamp locks

The variation in current measurements observed for clamp locks is also observable for other types of POE. The issue of a unique operational behaviour of each POE has also been raised by Tunley (Tunley, 2010). Such behaviour can depend on the age and frequency of usage of POE, configuration of the RPS, etc. Due to this diversity, it is therefore possible that fault-free movements of some point systems are similar to faulty movements of other point systems and therefore the same classification of such movements becomes likely. Thus data sets of individual point systems, not their groups, have to be analysed separately. Furthermore there are far more current trends from good movements than those from faulty movements. Thus multiclass classification algorithms that require similar size data samples for each class are of limited use in this case and a single class algorithm should be used instead.

2.1.6.3.2 Variable duration of POE operations

Varying durations of the operations of the same POE is the other feature observed in the in-field data, as demonstrated in Figure 2.7.

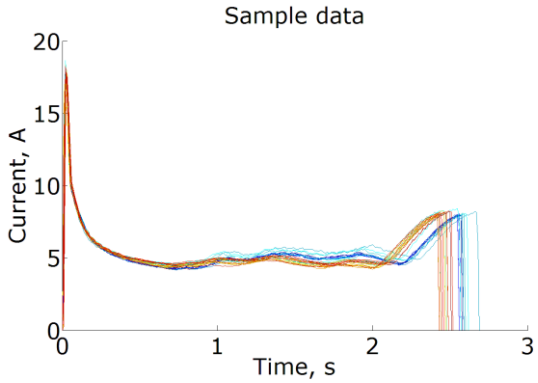


Figure 2.7. Measurements of current of clamp lock

As mentioned in section 1.1.2, the time series of measurements of current will be compared using similarity measures. However, the standard similarity measure, known as the Euclidean distance, cannot be defined for two time series of different length. In order to use Euclidean distance to compare two time series the following time series pre-processing approaches are the most common:

1. Adding trailing zeros to the shorter current time series to obtain the length of the longer time series.
2. Dividing the current time series into a fixed number of segments and calculating the mean of each segment.
3. Rescaling the time series to make them of equal length (Fu et al., 2008).

However, for the data available for this study, using the first method, differences between zero values and the available measurements of current would be of a big order (for example, values of measurements of current at the end of the switch rail movement are around 8-9 amperes) and thus it could significantly misrepresent the difference between two time series. Using the second case, a lot of information would be lost due to segmenting. Moreover, different phases of the movements might even be compared, giving us misleading information. As any two time series usually differ by a small number of data points available, an appropriate way of pre-processing the available data in this study seems to be the stretching or shrinking of the data to make the time series of equal lengths, as suggested in third approach. However, even after the rescaling some problems might arise, e.g. if the phases of the movements do not match in time, since the Euclidean distance allows only one to one matching between two time series.

Alternatively, there are several other similarity measures that allow elastic matching, i.e. allow the scaling of the time axis to compare two data points, which were measured at different time points. From such similarity measures, Dynamic Time Warping (DTW) (Sakoe and Chiba, 1978), Edit Distance with Real Penalty (ERP) (Chen and Ng, 2004), Time Warp Edit Distance (TWED) (Marteau, 2009) are the most commonly used ones. Due to the elastic matching, such methods can be applied without rescaling the time series. The drawback of such methods is that they are more time-consuming than the Euclidean distance method, since they calculate the differences of one point of the time series with all the possible points of the second time series. Thus their complexity is $O(n^2)$, while Euclidean distance complexity is $O(n)$, where n is the number of time series data points. Moreover, some parameter values of these algorithms need to be determined on individual case basis. These methods will be described in more detail in section 2.3.6.

Since POE current was the only observable parameter of the RPS, an accurate comparison of this parameter, i.e. a correct estimate of similarity measure values of this parameter is essential. Misleading results of similarity measures would result in the fault detection algorithm giving incorrect results (good movements classified as faulty and vice versa). Therefore, there is a need to investigate the suitability of using specific similarity measures in classification methods proposed for fault detection on railway point systems.

2.1.6.3.3 Changing behaviour of fault-free point system

Another feature unique to the infield railway point systems data that cannot be observed in the data obtained in the laboratory environment is the variability of current trends obtained during fault-free movements of the same point system. Such variability might be due to deterioration of the point system, weather conditions,

performed maintenance or some other causes. To illustrate this variability in the data available in this study, graphical analysis of one point system is given, but similar situations can be observed for all of the point systems. 5 current time series logged at the beginning of the observed period are plotted together with 5 current time series logged later in the observed period. This is done repeatedly to show how the current trends of fault-free point system are constantly changing. The movements from several different time intervals are plotted in Figure 2.8.

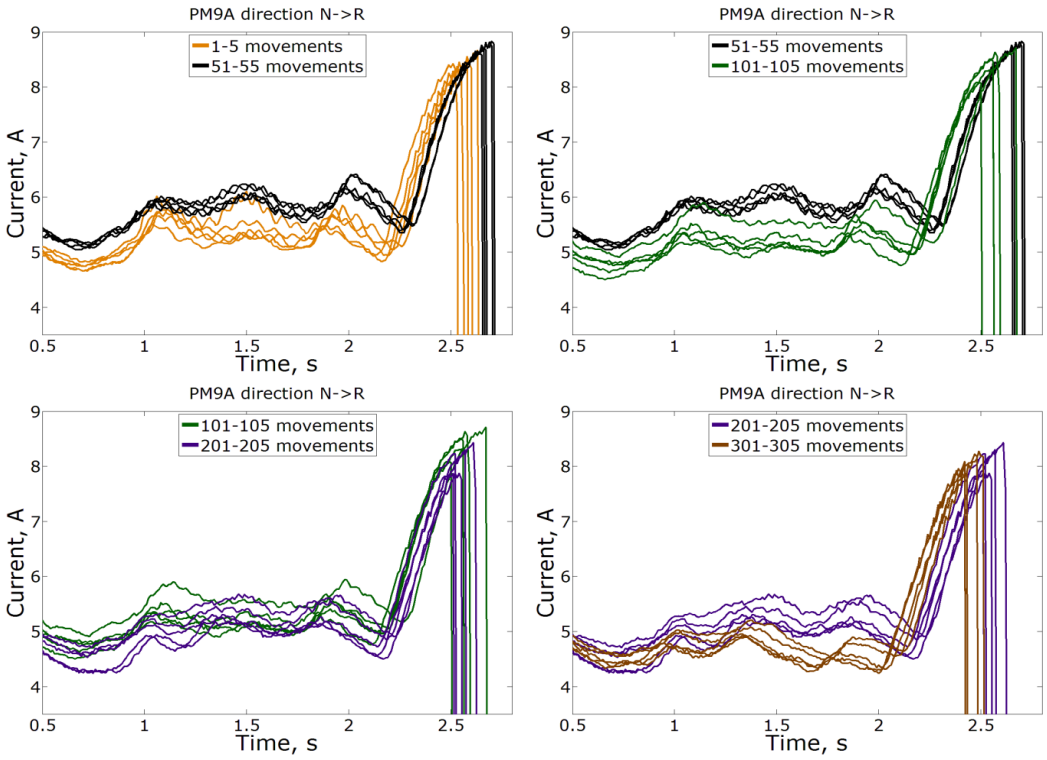


Figure 2.8. Changing behaviour of fault-free point system

The movements plotted in orange and labelled 1-5 (on the top left hand side of Figure 2.8) were performed during the first days of January. The movements plotted in black and labelled 51-55 (on the top left and top right hand side of Figure 2.8) were performed during the first days of February. The movements plotted in green and labelled 101-105 (on the top right and bottom left hand side of Figure 2.8) were performed in the middle of March. The movements plotted in indigo and labelled 201-205 (on the bottom left and bottom right hand side of Figure 2.8) were performed during the last days of May and first days of June. The movements plotted in brown and labelled 301-305 (on the bottom left hand side of Figure 2.8) were performed during the first days of August. One can observe from Figure 2.8 that the behaviour of fault-free system changed through time. The movements that were plotted in pairs in Figure 2.8 are also plotted on one plot in Figure 2.9.

A variety of current profiles can be observed from Figure 2.9. Such changing behaviour of POE during fault-free operation of points might cause difficulties detecting faults in the RPS.

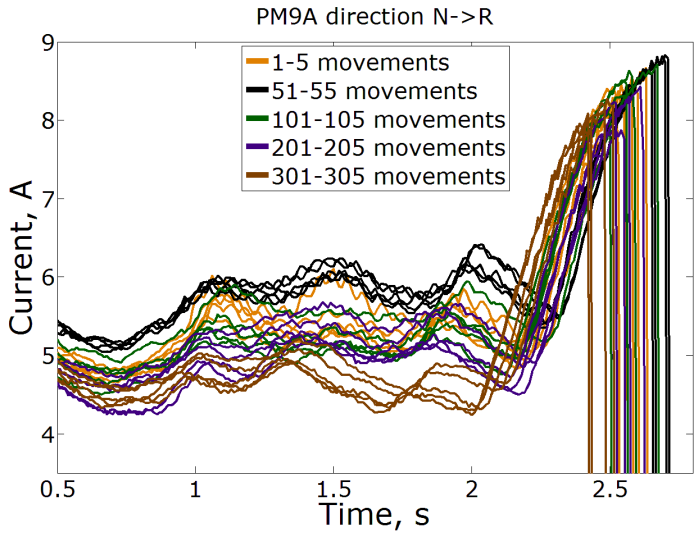


Figure 2.9. Variety of fault-free movements for one POE PM9A

A similar situation to the one discussed in section 2.1.6.3.1 can occur, when the diversity between current trends of two different POE might lead to a possibility that fault-free movements of one POE are similar to faulty movements of the other POE and therefore they can be misclassified. This time fault-free and faulty movements of the same POE might be more similar, if they were performed in a short time span between them, than two fault-free movements, which were performed with a longer time span between them. Thus, the changing behaviour of a fault-free RPS should be taken into account, when choosing an appropriate method for fault detection. In the next section the identification of faulty and good movements is presented.

2.1.6.4 Identification of faulty and good movements

Correct labelling of data used in a classification algorithm is of critical importance. Mislabelled data could lead to a potential decrease of the classification accuracy. Moreover, it could cause problems to find the optimal values of classification algorithm parameters, since these values are usually found by comparing the classification rates of the classification algorithms. Using the available data, the automatic labelling of POE operations data was done in two ways:

1. Label the movement data according to failure dates in fault records in FLD.

2. Label the movement data according to movement alarm dates recorded in MLD.

In the first case, dates of faults were used to determine, which movements were performed by faulty system. Based on these dates, the labelling was done in the following way: the movements recorded between the date of fault occurrence and fault rectification were labelled as faulty, other movements were labelled as good. However, in this case, one has to rely on the fact that the dates of fault occurrence and rectification have been recorded accurately in FLD, which might not always be the case.

If the data was labelled according to alarm dates, movements that raised an alarm were labelled as faulty and all the other movements were labelled as good. In this case, one has to assume that the alarms were raised only for the movements that correspond to the faulty (or deteriorating) condition of the point system, which again might not always be the case. If the labelling is misleading, poor classification

accuracy is likely to occur, independent of the algorithm chosen. Therefore, there is a need to investigate the suitability of the two labelling approaches.

2.1.6.5 Current profiles of failure modes observed in in-field data

In this section, the measurements of current of RPS with certain failure modes present in the database used in this study are described. The movements that were found to be faulty are plotted in red colour and the good movements are plotted in green. The same failure modes are analysed as those described in section 2.1.5.1, except for the tight lock failure mode, which was not found among the data that was available for the analysis.

Obstruction

As described in the previous sections, an obstruction in the railway point system should result in an increase of the current usage, when the obstruction is met and until the end of the operation. Depending on whether the obstruction can be forced out of the way, the railway point system is able or not to complete the operation. An increase of current usage can be observed in the example, taken from the movement database plotted in Figure 2.10.

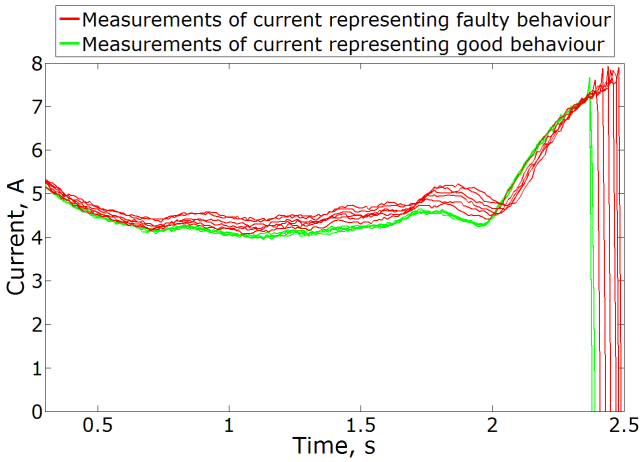


Figure 2.10. Profiles of current when there is an obstruction in RPS

In this case, the profiles of measurements of current indicate a presence of obstruction, which is only forcing the POE to use excess of current. The movement operation is completed, but its duration is longer than normal.

Dry or contaminated slide chairs

A similar situation can be observed for dry or contaminated slide chairs in Figure 2.11. The excess of current usage can be seen throughout the movement phase of the switch rails, from somewhere around first second of the movement. As in the case of obstruction, the faulty operations take longer to complete. This is due to a higher friction between the switch rails and slide chairs, since the slide chairs are contaminated or not properly lubricated.

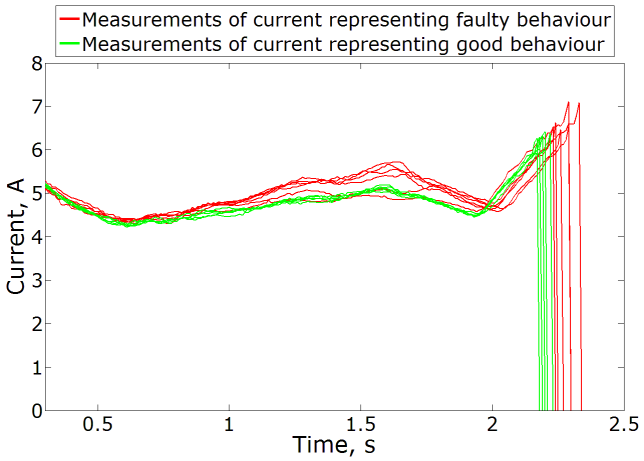


Figure 2.11. Profiles of current when the slide chairs are dry or contaminated

Back drive out of adjustment

The movements, which represented a back drive out of adjustment failure mode in the movement database, are given in Figure 2.12. An excess of current throughout the movement was expected, but it can be observed from the figure, that the excess of current is used only at the end of operation. Moreover, the motor of POE eventually times out after running for a long time. This profile is different from that expected to appear when a back drive is out of adjustment as described in 2.1.5.1.

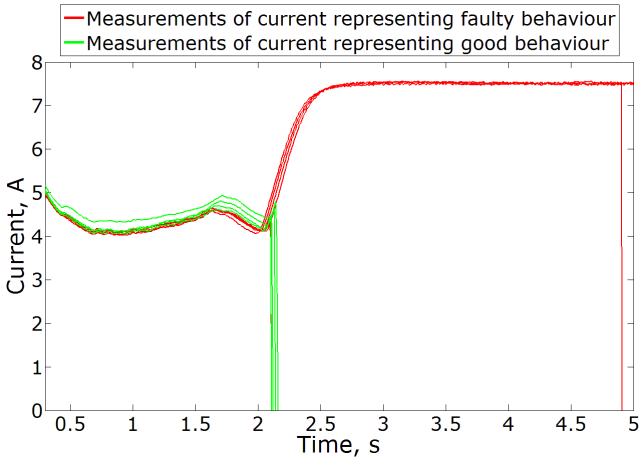


Figure 2.12. Profiles of current when the back drive was out of adjustment

2.1.7 Summary

An introduction to the railway point systems was given in this section. A diagram, identifying the main components of the railway point system, was given with a brief introduction of each component and how it affects the operation of the whole system. Failure modes of railway point system caused by components and external causes were then overviewed. Different maintenance regimes of railway point systems were presented, indicating advantages and disadvantages of each.

Analysis of system operation under different failure modes present revealed an increase of current used by POE motor for most of the cases. Thus the discrimination between different fault types using only current measurements is impossible in most of the cases and therefore fault diagnostics will not be considered in this study.

The main features of the data available in this study, which influenced the choice of suitable data analysis methods, were overviewed. Three main features of the data were identified: diversity of operational behaviour of the same type of POE, variable duration of the operation of the same railway point system and the changing behaviour of the fault-free point system.

Finally, an automatic labelling method of faulty and good movements was presented based on two types of data: alarm dates and failure dates. Both labelling options were used in this study and the results are presented in section 2.5.

In the next section a literature review of the state of the art techniques applied to railway point systems fault detection and prediction is given.

2.2 Literature review

2.2.1 Introduction

An extensive literature review has been performed in this study to identify the methods used for fault detection or prediction of the railway point systems. The variety of methods applied for fault detection shows that the research is still ongoing and one suitable method has not been found yet. In the next sections methods for fault detection and fault prediction are presented. The fault detection methods are grouped in the following categories: statistical analysis methods, classification methods, model based approaches and other methods. The fault prediction methods are presented individually as only several of them were found in literature.

A summary of each method is given in the following fashion, depending on the information available: introduction of the model, the failures considered, observed parameters for monitoring the condition of a railway point system, data obtained for the analysis, results obtained with the proposed method, advantages and disadvantages of the proposed method and a discussion whether it can be useful in this study.

At the end of this chapter, industrial solutions for fault detection of railway point systems are overviewed. A typical alarm threshold technique for fault detection is presented. Major disadvantages of this technique are illustrated with simple examples.

Finally, the key requirements for a new fault detection method that can accommodate the features of infield railway point system data (section 2.1.6) and deficiencies of previous methods found from literature review are formed.

2.2.2 Methods for fault detection of railway point systems

A wide variety of techniques have been used to develop a fault detection methodology for railway point systems. One of the most commonly used techniques is Principal Component Analysis (PCA). It has been used as a standalone technique and in a combination with some other technique, e.g. Support Vector Machines (Eker et al., 2012), Hotelling coefficient (Ardakani et al., 2012) or simply visual inspection (McHutchon et al., 2005, Márquez and Chacón Muñoz, 2010, Márquez and Garcia-Pardo, 2010). Other statistical techniques used included an analysis of squared sums of errors (Oyebande and Renfrew, 2002), a three statistical criterion technique introduced by Marquez (Márquez et al., 2003), a curve statistical parameters analysis (Zwanenburg, 2006), a qualitative trend analysis (Márquez and Schmid, 2007, Silmon and Roberts, 2010), a mixture discriminant analysis (Chamroukhi et al., 2008), a Kolmogorov-Smirnov test (Bolbolamiri et al., 2012) and a discrete wavelet transformation together with Support Vector Machines (Asada et al., 2013).

An expert system based on rules and thresholds was suggested by only one research team in (Atamuradov et al., 2009). Several model based approaches have been introduced, such as H_2 norm criterion based on a linear model (Zattoni, 2006), an unobserved component model (Márquez et al., 2007b), a state-space and harmonic regression models (Pedregal et al., 2009) and a vector auto-regressive moving-average model (Márquez et al., 2010). Only one researcher looked at the influence of external conditions of the railway point system, i.e. temperature and

weather conditions (Böhm, 2012). More details of these approaches are given in the next subsections.

2.2.2.1 Statistical analysis methods

Sum of Squared Errors (SSE) and Net Energy Analysis (NEA)

Oyebande and Renfrew (Oyebande and Renfrew, 2002) tried several different methods to detect RPS faults. Data was obtained from experiments with 3 point machines: a HW 2000 point machine in the laboratory, a HW 2000 point machine located in Manchester Metrolink's Queen's Road depot and a HW 2000 point machine located in the Undercroft of Piccadilly Station. The current used by point machine motors was measured by a Hull effect current transducer and the voltage was measured by a resistor. The motor speed was measured from pulses obtained by an inductive proximity detector, mounted adjacent to the step-down gearing stage on the motor shaft.

The authors were examining the performance of point machines when different loads were applied to the machines, rather than introducing distinct failures to the system. They based this idea on the findings that faults like a lack of lubrication, an obstruction or motor problems would reflect on the increased load of the point machine. Different load profiles were attained by adjusting the position of the piston attached to the tension spring with respect to the main cylinder with a screw thread. Several techniques were employed to discriminate between the point machine performances. Data from each point machine was tested separately.

The authors first performed a transient analysis of current waveforms by comparing the sum of squared errors (SSE) between them. In this test, the data was obtained by operating the point machine ten times at five different piston gap settings. Intuitively, the SSE values for the waveforms, which represent the operations with the same piston gap setting, should be smaller than for the waveforms which represent the operations with different piston gap setting. The SSE values were inspected visually by plotting them as a three dimensional surface. However, a clear trend has not been observed for the different loads applied neither for the laboratory point machine nor for the Queen's Road machine, used in this testing phase.

A spectral analysis of current waveforms was applied next. Spectral patterns were first extracted for the same data that was used for calculation of SSE values. The power spectral density (PSD) of each discrete signal was calculated. The SSE pairs were then calculated, replacing current waveforms with the corresponding PSD vectors of those waveforms. The results were compared in a same manner as previously, by plotting the SSE values in a three dimensional surface plot. Once more, the visual inspection indicated that there was a lack of discrimination between different operation conditions with different load profiles, and that the method could not lead to any valuable deduction.

The durations of switching operations were analysed by calculating the mean and standard deviation of movements in the normal direction. The duration of the switching operation was defined as a period of time between activating the power supply of the point machine and cutting the power supply off. Results obtained for the laboratory point machine showed that the mean time of all the switching operations was 1.71 s and the width of the deviation was 1.8%. Therefore this approach could only give good results in cases of a severe disruption of the POE operation, since the durations of the switching operations were very similar for different load profiles.

The analysis of the end-stroke positions was used to determine the problems caused by the lack of adjustment of the railway point machine. An end-stroke position was defined as a position of switch rails when they are locked after the switching operation. Trend analysis of the end positions over a period of time was performed using data acquired from 70 consecutive switching operations of the laboratory point machine. The positions of the switch rails were measured during the switching operation and that way a corresponding position waveform was obtained. Values of the position waveform registered after the completion of the switching operation were averaged. The averages of the end positions were then plotted on a scatter diagram with the best straight-line fits. After a visual inspection of the graph, this method was also concluded to lack the distinction between different loads applied to the point machine.

A more robust approach to identify the different load influence on the point machine behaviour proved to be the usage of at least two different types of waveforms, that is, measurements of current and voltage data. In that way random fluctuations in one measurement had less effect on the overall result. For that purpose, a Net Energy Analysis (NEA) technique was used, which involved the calculation of the net energy (EP) expended in moving the switch rails during one switching operation. Plotting the scatter of EP values showed a clearly distinguishable trend between different piston gap settings.

From all of the methods applied in this study, the NEA technique performed the best. The authors showed that there is evidence of different point machine behaviour when different loads are applied. However, the distinction between various load settings was done heuristically by looking at scatter plots of EP values. This is one of the biggest deficiencies of this approach, if considering using it as an online condition monitoring system. All other methods that were used by the authors failed to accurately distinguish between the different load settings. Thus, all of the proposed techniques from this research cast doubt on using them if one is interested in the automated detection of the RPS deteriorated or failed condition.

Statistical criterion testing

Marquez et al. (Márquez et al., 2003) carried out research on the fault detection of RPS using data of force (in newton) versus time (in seconds). They suggested analysing absolute values of the difference (equation **2.2**) between the actual data and reference data, because from the actual curves of force versus time, faults could not be detected directly.

$$\mathrm{d}_\mathrm{i}^\mathrm{j} = \left|\mathrm{x}_\mathrm{i}^\mathrm{j} - \mathrm{x}_\mathrm{i}^\mathrm{s}\right|, \qquad \textbf{2.2}$$

where x^j is actual data and x^s is the reference data given by:

$$\mathrm{x}_\mathrm{i}^\mathrm{s} = \frac{\mathrm{x}_\mathrm{i}^{\mathrm{s}-1} + \mathrm{x}_\mathrm{i}^\mathrm{j}}{2}, \qquad \textbf{2.3}$$

where $s-1$ is the previous reference data set, j is the new fault free data set, s is the new reference data set containing x_i – the data value at time-step i.

With no faults present the ‘as commissioned’ tests resulted in the curves of force difference d^j which were very similar.

The first criterion used for the fault detection involved checking whether the shape of the test data curve was irregular. If the irregularities were not sufficiently distinctive to

detect a fault with the first criterion, the maximum position t_{max}^{j} of the curve d^{j} was tested with an allowed margin of t_{mg}. Finally, if the first two criteria have not resulted in the detection of a fault, then a third criterion was applied. It checked whether the curve of force difference d^{j} was symmetric with respect to the maximum position, again with a margin of a given width:

$$\left(\frac{t_{max}^{j}}{T^{j}-t_{max}^{j}}\right)\sum_{n=0}^{t_{max}^{j}} d_{n}^{j} \approx \sum_{m=t_{max}^{j}}^{T^{j}} d_{m}^{j}, \qquad \textbf{2.4}$$

where t_{max}^{j} and T^{j} are the maximum position and the total time respectively, $\sum_{n=0}^{t_{max}^{j}} d_{n}^{j}$ and $\sum_{m=t_{max}^{j}}^{T^{j}} d_{m}^{j}$ are the areas under and above t_{max}^{j} position of curve d^{j}.

Any set of the test data that satisfied the three criteria was considered to be the result of a test without a fault and was defined ‘as commissioned’.

To increase the reliability of the last criterion, the authors employed a Kalman filter (Kalman, 1960) to smooth the curves. The Kalman filter approach was used to filter out the noise from the data, which improved results quite significantly. A total of 151 experiments were carried out, 79 in the reverse to normal direction and 72 in the normal to reverse direction, when 12 failure modes were introduced in the experiments. With the Kalman filter, authors could detect 100% of faults in the reverse to normal direction in the 79 experiments. Without the Kalman filter the result dropped to 97.33%. In the other direction, 97.1% of faults were detected when using the Kalman filter and without it the result dropped to 94.2%.

Very good results were obtained with the proposed approach in the experimental part of this study. However, in the study, it was not clearly stated how many movements represented a failed condition and what type of failure modes were considered. Therefore, it is difficult to judge the suitability of this method to our study.

Principal components analysis (PCA)

McHutchon et al. used signal processing techniques by analysing statistical and geometrical parameters of force data, applying wavelet-transform and Principal Component Analysis, to detect the faults of railway point systems in (McHutchon et al., 2005). Experimental data was obtained from a laboratory Type HW point machine. The current data was obtained using a current transducer in a non-intrusive way. The force data was acquired using a load pin inserted in a place of the bolted connection between the drive bar and the drive rod. 8 different faults were artificially introduced in the system:

1. Tight lock on the reverse side (sand on bearers both sides).
2. 12mm obstruction at toe on the reverse side.
3. Back drive slackened, toe end, left hand side.
4. Back drive slackened, toe end, right hand side.
5. Back drive tightened, heel end, left hand side.
6. Back drive tightened, heel end, right hand side.
7. Diode snubbing block disconnected.
8. Drive rod stretcher bar loose right hand side.

151 switching operations were performed to obtain the faulty and fault free data: 79 in the reverse to normal and 72 normal to reverse directions. Each fault was

simulated three times, resulting in 24 faulty movements with faults present for each direction. The following statistical parameters were used in the study to extract the information from current and force measurements: maximum, minimum, mean, peak-to-peak, standard deviation, root mean square, shape factor, crest factor, impulse factor, kurtosis and absolute area.

27 movements were chosen to test the method: 24 faulty movements and 3 ‘as commissioned’. Faults, numbered 2, 3, 4, 6 and 8 were detected by plotting the root mean square against maximum force values. However, this was only deduced by visual inspection and no automated way of detecting faults based on the statistical parameters was suggested.

In the next phase of the study, the original data was transformed with a discrete Daubeches wavelet transformation and then the same statistical parameters were obtained as those for the non-transformed data. A detailed algorithm of the Daubeches wavelet transformation can be found in (Daubechies, 1992). Once more, the extracted statistical parameters were plotted against one another and a visual inspection was performed to discriminate between the different failure modes. However, again there was no automation in the fault detection technique.

Lastly, the PCA method was applied in a similar manner. The idea of PCA is to convert original data, which is possibly highly correlated, into a set of principal components, which are linearly uncorrelated. The PCA method allows a reduction of the dimensionality of the data by transforming the data into a few principal components that reproduce the majority of the variability in the original data and thus is frequently used in practical applications to pre-process the raw data. Details of PCA can be found in (Pearson, 1901). Principal components were found from the statistical parameters. The principal components were then plotted against one another and a visual inspection was performed once more. Only three faults were detectable in this case: faults numbered 2, 6 and 8.

Some faults were detectable with the techniques proposed in this research. However, the distinction between different failure modes was finally done by visually looking at the plots obtained from applied methods. Thus neither of the approaches gives quantitative information on differences among failure modes and they could not be used if an automated fault detection technique was required.

Marquez together with Chacon Munoz (Márquez and Chacón Muñoz, 2010) and Garcia-Pardo (Márquez and Garcia-Pardo, 2010) further extended the research on PCA for point system fault detection. 476 experiments were carried out collecting time, force and operating current data. The current and force data were used in the experimental part for both normal to reverse and reverse to normal movements. The same failure modes and the same statistical features were considered as those in (McHutchon et al., 2005). 3 dimensional vectors as Principal Components were extracted from these statistical features using PCA. Data of 26 movements was then used to test the results obtained by PCA. As in the previous research, the fault detection was made by visually inspecting the plots of Principal Components. However, some faults were not clearly separable by the visual inspection, and as in (McHutchon et al., 2005) no quantitative analysis was performed to separate them. Similar results were also obtained in (Márquez and Garcia-Pardo, 2010), as the same idea of PCA was used. Thus, the main limitation of the proposed approach was the heuristic reasoning for the fault detection.

Hotelling T2 coefficient

Ardakani et al. combined PCA with a Hotelling T^2 coefficient for the identification of a health state of point machines (Ardakani et al., 2012). The authors investigated the behaviour of double point machines that consisted of two motors working consecutively. In this study, operation of RPS was segmented into 4 segments and each of them was analysed individually. The first segment corresponded to the operation of the first POE, the second segment represented the locking function of the first POE, the third segment represented the operation of the second POE and the fourth segment was the locking function of the second POE. Three categories of data inputs were considered for each segment: statistical features (area, crest factor, impulse factor, kurtosis, maximum, mean, minimum, shape factor, standard deviation and the location of the maximum point) of the measurements of current, statistical features of the difference between measurements of current of the reference signal and the input signal, and the energy of a signal (the product of current and the voltage). Ratios of the energy values for each segment and energy values were considered as data inputs to PCA.

The PCA technique was employed to pick out the features of the energy of a signal that are the most important and explain the most variation in the data. PCA reduced the number of features from 10 to 4: energy of fourth segment, ratios of energy between first and fourth, second and fourth and third and fourth segments. The Hotelling T^2 coefficient was then calculated based on the principal components picked out by PCA. The Hotelling T^2 coefficient is a multivariate generalisation of the Student's t test (Hotelling, 1931) and is used to test whether the mean value of a multivariate object is equal to a chosen mean value. It was used as a health indicator of a point machine in the following way:

- Point machine was considered as healthy at the beginning of the analysed period. The initial health state of the POE was attributed number 5.
- Hotelling T^2 coefficient was calculated comparing the next operation of the point machine with a baseline operation. The resulting Hotelling coefficient was subtracted from number 5 and a new health state was obtained.

Authors did not reason their choice of using number 5 as an indicator of a healthy point machine. The obtained health state values were analysed graphically and the authors concluded that there was a strong correlation between maintenance events and the drop in health state values before the maintenance occurred. However, no suggestion was given how the health state values should be interpreted to identify a deteriorated state of the point machine, except for stating that the health state value should be around 5.

Kolmogorov – Smirnov (K-S) test

Bolbolamiri et al. used a K-S test to detect faults of RPS that occur due to other reasons than point machine components (Bolbolamiri et al., 2012). Such failures might include slide chair friction, obstacle between the stock and the switch rails, damage to the rails etc. Authors assumed that when this type of a fault occurs, the mean value of the current measurement will change from its base value μ_0 towards some new increased value $\mu_1 > \mu_0$.

Two failure modes were analysed in this study: obstruction and dry slide chairs. The obstruction was analysed with two levels of severity: insertion of soft rock and hard rock. For the investigation of dry slide chairs, the slide chairs were degreased

manually. This type of failure reflected on the increase of current drawn by the point machine throughout the movement.

16 movements were sampled for the normal state and each failed state, giving total of 64 movements. For the K-S test (Massey, 1951) a reference signal (non-faulty movement) was used to decide whether a new movement was faulty or not. The idea of the K-S test is to compare empirical cumulative distribution functions of the two given samples and to test the hypothesis that the distributions are equal with a given significance level $\alpha = 0.05$. The p-values of the K-S test for the faulty movements, which were compared to reference (non-faulty) movements, should be less than the given significance level α. This would show that the empirical distributions of the measurements of current were statistically significantly different for faulty and non-faulty movements.

The approach was tested with 5 non-faulty and 15 faulty movements. Measurements of 4 out of 5 good movements were found to have the same empirical distributions with the given significance level $\alpha = 0.05$. The measurement of current of fifth good movement had small p-values (0.053, 0.075, 0.063) with first, third and fourth good movement measurements. One p-value, for second movement measurement of current, was even smaller than the significance level α (0.0496<0.05), indicating that the measurements of current of those movements had different empirical distributions. The empirical distributions of faulty and non-faulty movements differed in all cases. Thus, there was only one error when the empirical distributions of two good movements did not match, with the given significance level.

Results obtained using the K-S test were very good, however the authors stated that this approach assumes that the fault should strongly influence the empirical distribution function of the measurements in order for the fault to be detectable. Thus this approach probably would not be able to detect a deteriorating condition of the point machine, since the changes to the measurements of the current are not that abrupt as those in the obstruction and dry slide chairs failure modes, which were manually introduced in this study.

2.2.2.2 Classification methods

Mixture Discriminant analysis (MDA)

F. Chamroukhi et al. (Chamroukhi et al., 2008) considered the RPS fault detection as a pattern recognition problem. The proposed approach using signals of electrical power consumed during the switch actuation period consisted of three steps:

- Feature extraction from the signal;
- Learning of parameters for different classes from a labelled collection of signals;
- Classifying the new data based on the learned parameters.

Each electrical power signal consisted of 550 points sampled at 100 Hz. Each signal was parameterized with the help of a regression model with a hidden logistic process by a parameter vector of dimension 21 (7 parameters for each of the 3 central phases – unlocking, moving of the switch rails and locking).

According to the MDA approach, signals were grouped into classes, each class C_k density was modelled by a Gaussian Mixture distribution. The mixture distribution parameters for each class C_k were estimated by the Expectation-Maximization (EM)

algorithm. The optimal number of Gaussian distributions R_k for each class C_k was computed by maximizing the Bayesian Information Criterion (BIC). Each new signal designed by the feature vector x_i was assigned to the class k^* which maximized the posterior probability that x_i originated from the class C_k.

A database of 119 real signals with known classes has been used for the experimentation part: 90 signals for training and 29 signals for the evaluation of the classifier. Three different classes of signals were considered in this paper: C_1 – class without defect, C_2 – class with minor defect and C_3 – class with major defect. The classification accuracy obtained by the proposed MDA approach was compared with the accuracy provided by: the Neural Network (NN) approach based on a multilayer perceptron with a single hidden layer of 13 neurons, the K Nearest Neighbours (KNN) approach and the Bayesian discrimination (BD) approach based on a single Gaussian distribution per class.

The results in Table 2.1 show that the MDA approach outperformed the alternative approaches, as it classified 95% of the data correctly, while the best alternative approach (Neural Network) achieved 90% classification accuracy. However, the differences between the movements representing different classes had clear visual differences and thus this approach should be tested, when the data is more difficult to distinguish.

Table 2.1. Comparison of percentage of good classification

Approach	Percentage, %
MDA	95
NN	90
KNN	88
BD	75

Support Vector Machines (SVM)

Asada and Roberts proposed a new approach to detect faults of AC point machines (Asada and Roberts, 2011). They chose to analyse the faults that develop gradually (incipient faults) and are mainly caused by a misalignment of components. Two types of incipient faults were considered: under driving and over driving of the drive rod. Data was collected from an AC point machine in training facility at the Central Japan Railway Company in Tokyo. Drive force, electrical current and electrical voltage were the observed parameters used in the study.

Initial K-means clustering analysis of the parameters showed that the drive force data has the best potential to distinguish between the normal and faulty behaviour of the point machine. However, the drive force data is hard to collect in practice and the additional sensors needed might introduce additional failure modes and additional costs. For this reason, an additional parameter was derived from the electrical current and voltage data. This new parameter, called electrical active power, was used in the K-means analysis. The electrical active power performed slightly poorer than the drive force, but for practical reasons, the electrical active power data was used in the further analysis.

In the next phase of the study, Discrete Wavelet Transform with level 9 Haar wavelets was performed to extract features of the original waveform of the electrical active power. The idea of Haar wavelets is to decompose the original data into a group of square shaped waves that allow a representation of the original data in a reduced dimensionality. The extracted features were used as inputs to the SVM. Two kernels of SVM were used: linear and radial basis function kernel. The kernel of the

SVM is a function that transforms the input data into an inner product space, which allows the input data to be better classified. Both kernels performed equally good achieving a 100% classification accuracy.

More testing on the AC point machines has been performed in a later work by Asada et al. (Asada et al., 2013). Different health states of the point machines have been considered. Testing was done with three combinations of point machine health states:

1. Over driving and under driving of the drive rod versus non-faulty movements
2. Minor over driving and minor under driving of the drive rod versus non-faulty movements
3. Minor under driving versus under driving; non-faulty movements versus minor under driving; minor overdriving and overdriving.

In the first two cases the SVM was able deal with the classification problem perfectly, by achieving a 100% classification accuracy. However, in the last case, when minor under driving was compared to non-faulty movements, the SVM classified all the movements as non-faulty. Thus, a small under driving of the drive rod could not be separated from the non-faulty movements by the proposed approach.

The same methodology as proposed for the AC point machines was tested with the DC point machines by Asada and Roberts (Asada and Roberts, 2013). This time four different faults of driving rod were considered:

1. Left-hand over driving;
2. Right-hand over driving;
3. Left-hand under driving;
4. Right-hand under driving.

One versus one multiclass SVM approach with voting was chosen as a fault classification method. 142 movements were used to test the approach: 30 fault-free and other reflecting one of the 4 faulty conditions. 5-fold cross validation was used to test the performance of the multiclass SVM. 99.33% accuracy rate was achieved for this approach. The minor fault conditions as in the case of AC point machines were introduced to test the SVM abilities further. In this case, Radial basis function kernel outperformed the linear kernel of the SVM by achieving a 99.57% classification accuracy rate.

Eker et al. performed fault detection of RPS based on the measurements of linear ruler to detect the misalignment of drive rod (Eker et al., 2012). Statistical features were extracted from the measured ruler data: mean, standard deviation, variance, slope, maximum and minimum. 20 measurements in total have been made: 10 for the faulty class and 10 for the fault free class. Maximum and minimum values were selected as the most important features after the t-test. PCA was also applied on the statistical features. The features selected by t-test and principal components extracted by PCA were then used as inputs to the SVM. The effect of different inputs on the performance of SVM was tested based on the classification accuracy of SVM. SVM with Gaussian kernel was chosen for this analysis. The approach with feature selection by t-test gave 83.3% classification accuracy, while the approach with PCA outperformed the t-test by achieving 100% classification accuracy. However, the SVM model was built to identify only one type of failure – drive rod out of adjustment. The proposed approach needs to be tested with the data of more failure modes if it is to be used as a fault detection tool.

Neural networks

Thomas Böhm suggested using external data (e.g. weather conditions) for the improvement of the railway point systems fault diagnosis (Böhm, 2012). The author analysed the diagnostic capabilities of the industrial railway point system fault diagnostics solution, known as SIDIS W (Siemens AG, 2011). SIDIS W used the alarm threshold technique for the early faulty detection. For this purpose, effective power was measured throughout the movement of points system. Each movement was divided in several subsections, representing a specific operation of the RPS. Eight different indicators were used to assess the point system condition. If any of these indicators fell below or exceeded the predefined thresholds, an alarm was raised: yellow alert identified a smaller deviation from the normal condition and a red alert identified a big deviation from the normal condition. These alerts were indications of an actual or potential failure occurrence.

11 point systems were considered for the evaluation of SIDIS W performance. The amount of alerts raised by the SIDIS W system highly varied for different point systems, starting from as little as 3.83% of yellow alerts up to 68.44%. The same tendency was observed with the red alerts where the percentage varied between 0.49% and 33.07% of all movements. Interestingly, the biggest fraction of red alerts was for the point system, which had no failures, while the smallest fraction was for the point system which had 2 failures.

The issue of the alarm threshold technique was the high number of false alarms. Thus, the author looked for the additional information, which could increase the accuracy of the alarm threshold technique. Temperature was considered in this study, as one of the possible addition to the power data. The weather data was obtained from the weather stations that were not further than 10 km from the RPS. The point systems were clustered by the same constructional characteristics, e.g. the switch type, the curve radius, the type of sleepers etc. Then a Neural Network was used to learn the alert thresholds for the power data. Correlation between the indicators of SIDIS W system and temperature data was calculated and when the value of correlation was below -0.6 or above 0.6, the coefficient of determination was calculated. Then an average change of the indicator per °C was multiplied by the coefficient of determination, giving the correction value per each °C. The average temperature of 11.3 °C was used as a reference temperature. Any deviation from this temperature was adjusted by adding a correction value, calculated as described before, to the power data. A Neural Network was used once more to calculate the alerts for the adjusted data.

The amount of yellow alerts was reduced by 16.84% (from 16997 to 14134) and 91.83% for the red alerts (from 4724 to 386). Such decrease in alerts, especially the red ones, shows a good potential of the usage of additional data to improve the performance of alarm threshold technique. However, the inclusion of temperature did not detect any new failures. The author showed that this additional information, which comes with no extra cost (the weather information is accessible openly), significantly reduces the amount of false alarms. The proposed approach to include the temperature information has successfully reduced the number of false alarms. Thus, the temperature information could be used to reduce some of the variability of failure-free movements.

2.2.2.3 Model based approaches

H2 norm criterion with linear time-invariant model

Zattoni (Zattoni, 2006) suggested using an H_2-norm criterion for detection of incipient failures by using a linear time-invariant model. The H_2-norm criterion of the linear model transfer function matrix specifies the root mean square residuals of the model at a given time. The residuals are then commonly compared to white noise for fault detection. A root mean square value of the residuals that is significantly bigger than the expected root mean square value of white noise is considered as an indication of faulty behaviour. The author was able to model behaviour of the current of the point machine in the switch rail moving stage with a linear parametric model. However, only the linear part of the current data from a point machine was considered. Linear models cannot be applied to model the current of a whole movement of point machine, because it clearly acts non-linearly. Since all of the other phases of the point machine operation were not considered, more details of this approach are not given. For more details on how the linear part of the movement was modelled using this approach, the reader is referred to (Zattoni, 2006).

Bivariate integrated random walk model

F. Marquez et al. (Márquez et al., 2007b) implemented a model from the class of unobserved component models to assess wear in railway points. Authors referred to an earlier work (Márquez et al., 2003) that used differences between time series of measurements of current. A bivariate integrated random walk model was used by the authors to forecast the measurements of current or force. A correlation coefficient ρ between the modelled measurements of current or force was compared with a predefined high threshold value at individual points in time, since the correlation between similar movements should result in a high correlation coefficient value. If correlation value ρ fell below a threshold this was an indication of a lack of correlation with the reference current curve and therefore the curve would be classified as ‘‘faulty’’ if the reference current was non-faulty. The authors showed visually how the ρ values slowly degrade for faulty curves. However, the authors did not give a procedure of how an adequate threshold to distinguish between a faulty and a good curve should be chosen.

State-space and harmonic regression models

Pedregal et al. used a state space and harmonic regression model based approach (Pedregal et al., 2009) to tackle the problem of failure detection of point machines. The current signals were measured from sensors installed in the point machine as the point machine performed operations in both directions: normal to reverse and reverse to normal. Movement duration or operating time was also measured. The fault detection system proposed was based on the comparison of the expected and actual shape of the current drawn over time. The proposed fault detection algorithm can be summarised in these steps:

1. Determine which historical data to use. In the given example, 50 previous free from failure movements were used.
2. Forecast the duration (together with 95% confidence interval) of the next movement with state space model.
3. Forecast the current signal by using the harmonic regression model with a time varying period (a range of durations were considered from the 95% confidence interval). In this way forecasts become available for a time horizon long enough to cover a full movement of the point mechanism.

4. Compare new data points measured by the sensors to all of the forecasts produced in step 3. The best forecast is chosen as the one with the minimum of the standard deviation of the error between the forecast signal and actual data.
5. If the best forecast from step 4 has a bigger standard deviation value than a pre-specified threshold value, then system issues a warning that a fault has occurred.

The algorithm was tested with a dataset of 380 movements in the normal to reverse direction, where 8 movements were abnormal and 380 non-faulty movements in the reverse to normal direction. The failure modes that were considered in this analysis reflected on longer operation of the point machine (operation takes 4 seconds instead of usual 2 seconds). The best discrimination between faulty and non-faulty movements was found experimentally with a standard deviation threshold of 0.4 units. No false alarms were raised and no faulty movements were missed. However, the proposed technique was not tested on the failure data, when differences between the faulty and non-faulty movements are not that severe.

Vector Auto-Regressive Moving-Average (VARMA) model

Marquez et al. (Márquez et al., 2010) used a similar approach to that of Pedregal et al. in (Pedregal et al., 2009), where the SS model was replaced by the vector auto-regressive moving-average model. The vector auto-regressive moving-average model (VARMA(p, q)) was used to forecast the duration of the movement. To be able to apply the VARMA model, the time series should have a stationary mean and variance. This is gained by taking the first differences of the signals. Multivariate autocorrelation and multivariate partial autocorrelation functions were used to identify the orders p and q of VARMA(p, q) model. For this particular dataset, these were found to be $p = 0, q = 1$.

The testing conditions of the algorithm were identical to the ones used in (Pedregal et al., 2009). As in the previous case, all the abnormal movements were identified and no false alarms were generated. To test the on-line capability of the system to detect faults, a recursive estimation of the standard deviation of the forecast signal error was obtained. If a fault was present, the standard deviation increased with time and when it reached a predetermined threshold value, a fault was detected. However, as in the previous study (Pedregal et al., 2009), the fault detection algorithm was tested only on the data, where failures were severe and showed a big increase in the duration of the operation (approximately 4 seconds instead usual 2 seconds). Thus such fault could also be detected using a simple threshold technique. In order to prove the validity of this model, more testing should be done on faults that deviate from the normal behaviour in some other way rather than the duration of the operation.

2.2.2.4 Other methods

Expert rule based decision systems

In their work, F. Marquez and F. Schmid (Márquez and Schmid, 2007) focused on the data received from a current sensor installed on a point machine motor. The obtained measurements of current were noisy, thus there would be a problem of a high number of falsely detected failures. The authors suggested using a Kalman filter (Kalman, 1960) to smooth and remove the irrelevant noise from the data. The parameters for the Kalman filter were found experimentally.

In the study, several extracted amplitude or time values from the measurements of current were compared to ”normal” values: the current absorbed during switch rails movement, the locking force expressed as current, the time elapsed from lock engagement until power off and the symmetry of three-phase currents. A rule-based detection of faults was performed using these extracted values. However details of the rules used for the detection of faults were omitted in this paper.

The fault detection algorithm was first tested by adjusting the measurements of current signals by superimposing distortions of +0.05A and +0.1A in a selected time interval to simulate mechanism faults. Random noise was added to fault-free signals. 100 simulations of fault detection algorithm were run to detect this manually introduced distortion of the measurements of current. Simulations showed that, even after using the Kalman filter, the accuracy of fault detection was 29%, when 0.05A increase in the measurements of current was introduced. When a bigger increase of 0.1A was inserted in the measurements of current, the accuracy of fault detection was 88% with Kalman filter and 50% without filtering. After this simulation, data from laboratory point machines was used to test this approach. The obtained data was the same as that used in several previous studies (Márquez et al., 2003, McHutchon et al., 2005). The fault detection accuracy in the reverse to normal direction was 100% with Kalman filter and 97.33% without filtering. In the normal to reverse direction the corresponding results were 97.1% with filtering and 94.2% without.

The fault detection results achieved in this study were very good. The authors showed how the filtering of data with a Kalman filter could improve the accuracy of the fault detection. However, little information was given how the actual fault detection was performed. Furthermore such a technique usually suffers from a big falsely detected failure level. An extensive expert knowledge is also required to build the rules for the fault detection. Therefore, if the condition of the point machine degrades over the time, even if no faults occur, the decision values for the rules would have to be changed to reflect the degraded condition.

F. Marquez et al. (Márquez et al., 2007a) improved the results of research in (Márquez and Schmid, 2007) by employing a moving average filter instead of Kalman filter for raw data filtering. This time, more parameters of point machine operation were used for the analysis: supply voltage of the point machine, current drawn by electric motor of point machine, current drawn by the whole system and force in the drive bar. After the raw signals have been smoothed with a moving average filter, comparison of the faulty and fault-free signals was made. However, the details of how the fault detection decision was made were not given in the paper. The authors only stated that differences between the observed parameters were calculated and were used in a pattern recognition model.

24 failure modes were tested in both the normal to reverse and reverse to normal directions of the operation. All of the faults were identified from the voltage, force and current drawn by the point machine data for normal to reverse operation. 16 out of 24 fault modes were identified from the current drawn by the whole system. Similar results were obtained for reverse to normal operation. All fault modes were identified from current drawn by the point machine and force data, but 19 of 24 fault modes were identified from the voltage data and 18 of 24 from the current drawn by the whole system.

It is hard to evaluate the actual performance of the proposed approach with such detail of the results presented in this paper: there was no information provided on how many measurements were used to represent each failure mode or the fault free mode, what was the accuracy of the fault detection. The authors just indicate whether

the faults could be identified from the observed parameters. Moreover, no information was given on whether there were any false alarms.

V. Atamuradov et al. (Atamuradov et al., 2009) used a time series analysis combined with an expert system for detecting faults. Data was first pre-processed using the smoothing of a moving average. After the noise was removed from the time series data, the time series were compared with the control signal, which represented the normal behaviour of the railway point system. The time series were divided into phases as in (Márquez and Schmid, 2007). A similarity measure between two time series was determined by the Dynamic Time Warping algorithm: the optimisation process was performed using dynamic programming to find the warping that minimised the area between the curves in time.

An expert system was then chosen to be simple decision rules to identify failed and healthy modes. Every decision rule had a boundary value, by which faulty modes were classified. Three different modes (two failure modes and a healthy mode) were modelled. Progression of a failure mode from the healthy state was simulated by calculating the weighted average of corresponding healthy and faulty time series data.

This method worked well when the sample data was not noisy, identifying healthy and failure modes with 100% accuracy. The increase of the noise in the data led to the worse results of identification of the fault, giving the accuracy of round 80% for both failure modes. All of the data representing the healthy mode was still classified with 100% accuracy.

Fuzzy rules with qualitative trend analysis

J. A. Silmon and C. Roberts (Silmon and Roberts, 2010) used qualitative trend analysis (QTA), to isolate the common characteristics of waveforms of measured parameters (power or current drawn by point machine motor) and form rules, which would describe the behaviour of the measured parameters under faulty conditions. The idea of the proposed method was to turn a waveform of measured parameters into a sequence of shapes, which would describe the qualitative and quantitative characteristics of the waveform. The waveforms were partitioned into equal segments and, based on the shape of that partition, a letter was assigned. The letters for representing the shape of the curve are shown in Figure 2.13.

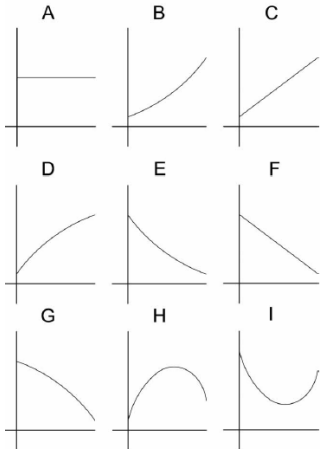


Figure 2.13. ‘Alphabet’ for representing the shapes of measured waveforms

The consecutive parts of the waveforms that had the identical letters were concatenated to make up an episode. The start and end values of the observed

parameter together with the number of the identical letters were added to each episode to represent and quantify the whole shape of the waveform of the measured parameter.

The proposed process to obtain the data for the training and testing phase of the analysis can be described in several steps:

- Simulate all the faults as well as a control test with no failures for each point machine. This dataset is called a test actuator (TA) dataset.
- Collect a dataset of fault-free data for each point machine. This dataset is called a fault-free monitored actuator (FFMA) dataset.
- Measure a new set of data for testing for each point machine. This dataset is called an operational monitored actuator (OPMA) dataset.

The differences between the episodes of fault-free (FFMA dataset) and faulty signals (TA dataset) were used to construct a set of fuzzy rules for the detection of each fault.

The data to test the approach was obtained from three laboratory based HW point machines: two with a back drive and one without. The measured parameters during operation of the point machine were current, force and displacement. However, the authors did not indicate which measurements were used to test the approach. The faults that were considered in this study were due to misalignments of the main drive or back drive (if the point machine had one): main drive overdriven to normal, main drive overdriven to reverse, back drive overdriven and back drive underdriven.

The fuzzy rules created in the training phase were applied to every measurement in the OPMA dataset, to compare the measurements with the ones in the FFMA dataset giving a membership value to each fuzzy rule. Over time, a trend of membership values was observed. An increasing trend of membership values for a specific fault indicated a potential failure.

When faults were introduced, the episode sequence (in terms of shapes) remained largely the same, but the quantities associated with each episode changed. The fault detection accuracy for both directions (normal to reverse and reverse to normal) was tested on two scenarios:

1. System must detect that the simulated fault is increasing in strength.
2. System must diagnose the fault by showing that the simulated fault is the strongest.

The results were measured by successfully fulfilled scenarios for different combinations of point machines used to obtain and test the fuzzy rules. For the main drive faults, one point machine (out of three) was used to obtain the fuzzy rules, while one out of remaining two was left to test the chosen scenario. This way, there were 6 possible combinations to obtain and test the fuzzy rules. For the back drive faults only two combinations were possible since there were only 2 point machines with a back drive.

Dealing with the correct identification of underdriven back drive was the biggest difficulty for this approach (3 out of 8 successful scenarios). The best effectiveness was achieved for the fault free and main drive overdriven to normal conditions, with the perfect identification of both scenarios. However, for the other type of conditions, the obtained results were worse, especially for the failures of back drive (9 out 16 successful scenarios). The authors indicated that such results were expected since

the failures of main drive were of bigger magnitude than the failures of back drive when the visual comparison of measured parameters was performed.

Thus this method might only be used for the detection of faults, where the failure condition significantly differs from the failure-free state. Furthermore, the information provided in this study did not indicate the number of operations which were successfully identified as faulty or fault-free. Such information would give a better idea how the method performed in terms of true fault detection and false alarms.

2.2.2.5 Summary

Various techniques and methods have been applied by researchers in order to detect faults of railway point systems. Some of them, namely three statistical criterion (Márquez et al., 2003), mixture discriminant analysis (Chamroukhi et al., 2008), state-space model (Pedregal et al., 2009), vector auto-regressive model (Márquez et al., 2010), Kolmogorov-Smirnov test (Bolbolamiri et al., 2012) and Support Vector Machines (Asada et al., 2013) performed well, detecting over 90% of faults. However, most of these approaches that achieved more than 90% fault detection accuracy were tested only on small datasets, e.g. 29 movements in (Chamroukhi et al., 2008), 20 movements in (Eker et al., 2012), 64 movements in (Bolbolamiri et al., 2012). The biggest dataset was combined from 380 movements, however only 8 of them represented the faulty condition (Pedregal et al., 2009, Márquez et al., 2010).

Moreover, the data for the experiments were either obtained from laboratory point machines (Márquez et al., 2003, Asada et al., 2013), or by manually introducing the fault to the system. Such data might not represent actual operating conditions of the point systems well which is influenced by extreme weather conditions or big forces from the passing trains. Furthermore, the behaviour of the railway point system changes with time and this factor has not been taken into account in the previously discussed approaches. Recent research by Thomas Böhm showed that external information, such as temperature, could be a likely reason to explain the varying behaviour of good movements, presented in section 2.1.6.3.3 (Böhm, 2012).

Data-based methods seem to be more suitable than the model-based methods for the fault detection of RPS. Support Vector Machines (Asada and Roberts, 2011, Eker et al., 2012, Asada et al., 2013, Asada and Roberts, 2013) was a widely used example of data-based methods. In one example, the combined PCA and SVM approach (Eker et al., 2012) was applied to the data of linear ruler, which only measured the position of rail during the movement. This sensor is capable to detect specific faults, namely only the ones that are related to any misalignment in the system. The approach of using SVM proposed by Asada and Roberts (Asada and Roberts, 2011, Asada and Roberts, 2013) achieved good results with classification accuracies close to 100%, however the data that was used in the analysis was again obtained from system operating in the laboratory.

A procedure to segment the movements of RPS with the required precision of regression models was given in (Chamroukhi et al., 2009). The coefficients of the regression models were used to classify the movement as belonging to class with or without a defect, obtaining 94% classification rate. Results obtained using the classification algorithms with real data were very good, and their effectiveness could be further tested when the trends between the classes are difficult to distinguish.

Most of the research has been done on the fault detection methodology and only a few papers have been published where authors suggest some automated fault

diagnostic capabilities of their methods (Asada and Roberts, 2011, Asada et al., 2013, Asada and Roberts, 2013). SVM was the technique considered in these three different publications. However, as mentioned before, small datasets were used to validate this approach and thus more testing could be done to test the efficiency of the method.

The key finding from the literature review was that most of the methods used for fault detection were tested on data, which did not have the features observed in the in-field data used in this study as discussed in section 2.1.6. The data features observed in this study are more challenging than the features of laboratory data, where variety of good movements is very little, the labelling of the data is perfect, etc. Moreover, most of the techniques presented in this section need a good representation of faulty data and need to be trained with both good and faulty data. However, in practice, the amount of failure data is very small and usually significantly smaller than the amount of data representing a fault-free condition. Thus the training of the model on both good and faulty data cannot be an option for each RPS. The research presented in this study is trying to fill such gaps by taking into account all the features of infield railway point systems data.

The methods for fault prediction of railway point systems are reviewed in the next section.

2.2.3 Methods for fault prediction of railway point systems

In the previous section approaches for fault detection of railway point systems were overviewed. Nevertheless, the ability to detect faults does not prevent their occurrence and possible safety issues and financial losses. Thus, detection of system degradation and fault prediction are highly desirable as repairs can be scheduled, avoiding major disruptions to train services (Márquez et al., 2008). However, the fault prediction problem has not been widely researched (Yilboga et al., 2010, Eker et al., 2011, Chamroukhi et al., 2011). The approaches proposed in the literature are reviewed in the following part of this section.

Time Delay Neural Network

H. Yilboga et al. (Yilboga et al., 2010) focused on predicting point system failures instead of identifying their occurrences after they already had happened. In their work they used a time-delay neural network approach to predict the Remaining Useful Life (RUL) of a point system. A Time Delay Neural Network (TDNN) is a neural network that incorporates time information in the network structure. The structure of a TDNN is set so that the input of the Neural Network consists of parameters at different time points. Thus, the effect of past values of parameters can be incorporated into the problem. The input can also include the past outputs of the TDNN. The Levenberg-Marquardt (Levenberg, 1944) learning method was applied to train the TDNN.

The degradation level of the system was modelled with an exponential degradation model. The result of the exponential degradation model was then converted to discrete degradation levels (health states) as shown in Figure 2.14, where different curves represent different degradation paths modelled by the exponential degradation model. Then, the TDDN is used to predict the future degradation levels of the system using past degradation levels. The number of predictions of health states made until the health state reached a predefined failure threshold is defined as RUL.

The RUL predicted values were compared with the real RUL values for 10 turnout systems. The results are given as a mean of the coefficient of determination $E(R^2) = 0.92$ and a mean of root mean square error $E(RMSE) = 2.75$. The mean of the coefficient of determination, $E(R^2)$, is high, so there is only 8% (on average) of data variability not explained by the model.

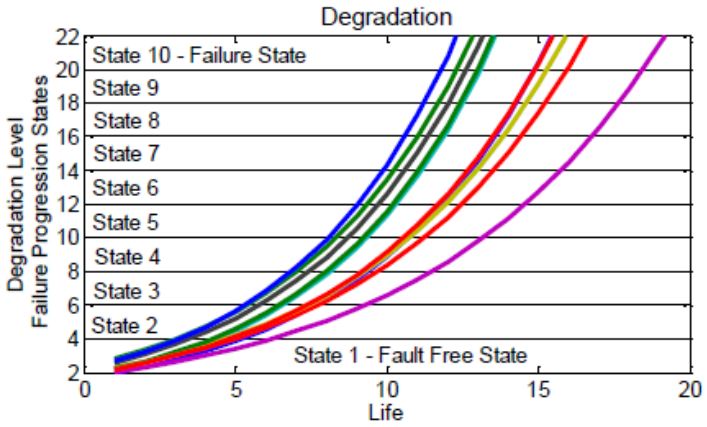


Figure 2.14. Degradation paths conversion to degradation levels

Simple State-Based Prognostic model

Eker et al. extended the work by H. Yilboga et al. in RUL prediction (Eker et al., 2011), where a Simple State-Based Prognostic (SSBP) model was presented. The same degradation model was used as in the previous work. A failure was defined at a point when the health state reached a predefined threshold.

The SSBP model included three steps: clustering of the continuous health states to determine the discrete degradation levels, cluster evaluation in terms of the optimal number of clusters and expected RUL calculation. For clustering, data from the different health states of multiple systems was clustered using k-means algorithm. The Calinski-Harabasz (CH) cluster validity index was chosen as the most robust one for the data set used. The optimal number of clusters, and thus health states, was chosen as the one that had the biggest CH index value.

The expected RUL was calculated using the transition probabilities between health states. The current health state of the system, h_c, was obtained using the clustering method. The probability of each value of the RUL between 1 and the RUL upper limit was calculated one by one. Only transitions from one health state to itself or to a consecutive state were allowed. The probability of staying in health state h_c for st_{h_c} time intervals means that the health state transitions to itself $st_{h_c} - 1$ times and then transitions to the consecutive health state one time. The RUL was defined as follows:

$$[RUL|st = h_c] = \sum_{i=1}^{RUL_U} i \times \prod_{n=h_c}^{h_f} q_{n,n}^{st_n - 1}\left(1 - q_{n,n}\right), \qquad \textbf{2.5}$$

where h_c is current health state, i is the current time moment, RUL_U is the upper limit of RUL, h_f is the failure time, $q_{n,n}$ is the probability that current health state n does not change and $1 - q_{n,n}$ is the probability that the health state transitions from n to $n + 1$.

An electromechanical type turnout with two drive rods, one for each rail, was used to collect the data for analysis. Force sensors were used to measure the tensile and

compressive forces. Current sensors measured DC current levels. Time series data was acquired for both normal to reverse and reverse to normal movements of the turnout system. A dry slide chair failure mode was considered in this paper, which was obtained manually. To obtain this failure state, the slide chairs have not been lubricated for a long time. Collection of the data was then carried out when oiling slide chairs one by one from the farthest to the closest slide chair of the turnout system until all of the slide chairs were oiled. The final state represented the fault-free state.

Samples were divided into 80% for the training data and the rest were used for the testing data. The CH cluster validity index obtained from training data indicated that 8 health states should be considered when modelling the failure progression. The authors compared this approach with a hierarchical hidden Markov model and showed that the SSBP outperforms the Markov model. The results were presented in a same manner as in previous work, by giving the mean of coefficient of determination $E(R^2)$ and the mean of root mean square error $E(RMSE)$. To differentiate between the Hierarchical Hidden Markov and Simple State-Based Prognostic models, they have been noted as HHM and SSBP:

$$E(R^2_{SSBP}) = 0.96; E(RMSE_{SSBP}) = 0.9;$$
$$E(R^2_{HHM}) = 0.88; E(RMSE_{HHM}) = 1.35.$$

The authors managed to increase the coefficient of determination of their model to $E(R^2_{SSBP}) = 0.96$. The downside of this approach is that the results were compared to RUL data from a modelled degradation process, so more testing needs to be done using real degradation data.

Switching autoregressive model governed by a hidden logistic process

Chamroukhi et al. extended their research on the hidden logistic process (Chamroukhi et al., 2011). Data of power curves was pre-processed in the same way as in (Chamroukhi et al., 2009), with the help of regression models. In this way, a parameter vector of 33 dimensions for each movement was obtained. Then the observation sequence $(y_1, \dots, y_T)$ was assumed to be generated by the multivariate autoregressive model governed by a stochastic process $(z_1, \dots, z_T)$.

The stochastic process $(z_1, \dots, z_T)$ controls the switching from one autoregressive model to another from among K models. It differs from the basic models in such a way that it has time dependent model parameters and covariance matrices. That way the non-stationary behaviour of the time series can be captured. Process $\boldsymbol{z} = (z_1, \dots, z_T)$ was assumed to be logistic. The conditional probability, π_k, of each state $k = 1, \dots, K$ is given by:

$$\pi_k(y_{t-1}; \boldsymbol{w}) = p(z_t = k | y_{t-1}; \boldsymbol{w}) = \frac{\exp(\boldsymbol{w}_k^T y_{t-1})}{\sum_{j=1}^{K} \exp(\boldsymbol{w}_j^T y_{t-1})}, \qquad \textbf{2.6}$$

where $\boldsymbol{w}_k$ is the parameter vector associated with k-th logistic process component z_k.

The proposed model was parameterised by a parameter vector $\Psi = (\boldsymbol{w}, B_1, \dots, B_K, \Sigma_1, \dots, \Sigma_K)$. The Expectation – maximisation algorithm was used to estimate these parameters.

This approach could be used for current health state identification of a POE or even the prediction of a future state. The current state can be identified by using the maximum a posterior rule:

$$\widehat{z_t} = \arg\max_{1 \le k \le K} p\left(z_t = k \middle| y_1, \dots, y_t; \widehat{\Psi}\right), \qquad \textbf{2.7}$$

where $p\left(z_t = k \middle| y_1, \dots, y_t; \widehat{\Psi}\right)$ is the posterior probability of the state k for the new observation y_t given observation y_{t-1} and model parameters $\widehat{\Psi}$. The prediction of the future state z_{t+1} given historical data up to t is performed by maximizing the prediction probability with respect to k:

$$p\left(z_{t+1} = k \middle| y_1, \dots, y_t; \widehat{\Psi}\right) = p\left(z_{t+1} = k \middle| y_t; \widehat{\Psi}\right) = \pi_k(y_t; \boldsymbol{w}), \qquad \textbf{2.8}$$

which is given by equation **2.6**.

The proposed model was tested with real data in terms of future state predictions as in equation **2.8**. 120 observations were obtained with 3 different states as in (Chamroukhi et al., 2009): $k = 1$ no defect state, $k = 2$ minor defect state, $k = 3$ crucial defect state. For this data, the state prediction error rate was 10.92%. The proposed approach gave very promising results for the prediction of the state of the railway point system. The effectiveness of the approach could be further tested when the trends between the three classes are difficult to distinguish.

2.2.3.1 Summary

Yilboga et al. and Eker et al. proposed a novel technique to predict the RUL of point systems by identifying their health state (Yilboga et al., 2010, Eker et al., 2011). A TDNN method has been used to predict the RUL in (Yilboga et al., 2010) and SSBP model in (Eker et al., 2011). The testing of these two approaches showed very good results, when the predicted RUL was compared with RUL from the exponential degradation model. TDNN approach predicted RUL with $R^2 = 0.92$ and the SSBP approach with $R^2 = 0.96$. These coefficients of determination show that the predicted RUL explains 92% of the real RUL variation in the case of TDNN approach and 96% in the case of SSBP.

However, the downside of these approaches is that they were tested on a modelled degradation path, which might be very different in the real system. The other downside of the TDNN approach is the choice of the optimal health states. Outputs of this degradation model were divided into discrete levels with a chosen number of health states determining the current health state of the point system. If too few health states were chosen, information was lost and the progression of degradation from one health state to another was too rapid. Furthermore, predictions based on too few health states might be over predictive (predicting RUL longer than the actual RUL). This issue has been dealt within the SSBP approach in (Eker et al., 2011) by finding the optimal number of health states. However, once more the validation of the model was done using the modelled degradation process, thus applying this method to real data could be misleading.

A better approach was proposed in (Chamroukhi et al., 2011). The authors extended their research from fault detection to fault prediction by using autoregressive models with a hidden logistic process. The approach was tested on real data of railway point machines. The state of the railway point machine was predicted with a 10.92% error. However, as most of the approaches, this approach was tested on data where the variability of good movements was low and the difference between faulty state and fault-free state was clear.

2.2.4 Industrial solutions for fault detection of railway point systems

2.2.4.1 Overview

There are a number of solutions available in the industry that offer monitoring services for railway point systems: starting from sensors and logging equipment and finishing with software solutions for fault detection and even prediction. Some of the solutions are based on measurements of the current (POSS from Struktron Systems (Strukton Rail, 2013)), Sidis W from Siemens (Siemens AG, 2011)), others are based on measurements of power (SPX Point machine monitoring system (SPX Rail Systems, 2013)). The last group measures a lot of different parameters, including current, power, temperature, displacement etc. including CDSRail Point Condition Monitoring (CDS Rail, 2013) and Integrated Remote System for Monitoring Point Machines from Thinking Forward (Thinking Forward, 2013)). A novel approach was suggested by Struktron Rail to inspect the points with a video inspecting train.

Most of these industrial solutions are based on the threshold alarm systems for fault detection and they also allow users to visually compare the trending values of the observed parameters. Alarm levels usually have to be specified manually and might not always detect changes in the point system operations, which might lead to a failure.

2.2.4.2 Typical condition monitoring system based on alarm threshold technique

An online condition monitoring system used for RPS commonly consists of several parts: sensors, loggers, modems and servers. Usually, current data is collected from the POE via a non-intrusive system, e.g. current transducers can be fitted to the POE motor feed cable so they do not affect the operation of the equipment under observation. Loggers are used to collect the data from the POE. It could be current versus time measurements, the time and the direction of the movement. The data is further passed to a server that allows engineers to view the data and measurement trends. Simple alarm thresholds for fault detection are usually implemented in the server. The values of the alarm levels must be tuned from time to time due to seasonal changes and the wear-out of the equipment. Two alarm levels are defined: low alarm with a lower priority, meaning that the changes in the observed parameter are of small magnitude but require a closer look at the trends of the parameter and high alarm, which indicates a significant deviation from normal operation, requiring attention from the maintenance personnel.

Commonly, such an approach based on threshold values is incapable of detecting a fault at its earliest stage. Engineers could benefit from the early detection of the incipient fault in several ways:

- Improve the degraded condition of the railway point system before the actual failure occurs.
- Plan the maintenance of the railway point systems and prioritise the maintenance of the worst condition systems.

This way, the chance of delayed trains would be minimised, since the condition could be improved when the traffic is low. The unexpected failure, by contrast, would result in speed restrictions on the line or even the closure of the whole line, leading to delays.

In a condition monitoring system based on the alarm threshold technique the major role is played by a correct threshold setting. If one uses too high threshold values the deteriorating condition of the system, e.g. increases in current, might be undetectable. On the other hand, if one uses very low threshold values a lot of false alarms can be raised for good movements. Usually the alarms are set for three types of variables: average and peak values of the current used throughout the movement of the point system and the motor running duration. Separate thresholds are set for each of these variables and the raised alarms indicate which of them exceeded the predefined thresholds.

The main deficiency of the alarm technique is that it does not take into account any changes in the shape of the measurements of current and thus only detects the failed condition when there are some severe changes in the data. Thus, this type of condition monitoring does not allow detecting the degrading condition of the RPS. Several simple examples are given to illustrate some of the deficiencies of the alarm threshold technique.

In the first case, the measurements of one good movement are taken which are then replicated with some modifications:

- Concentrate the values in a small area near the end of the movement, between time points 182 and 222 (plotted on the left hand side of Figure 2.15). This might indicate a problem in the locking phase of the point system.

- Concentrate the values in a wide area throughout the movement, between time points 95 and 222 (plotted on the right hand side of Figure 2.15). This could be just a small variation in the non-faulty behaviour of the railway point system.

The same average values of replicated movements are obtained in both ways.

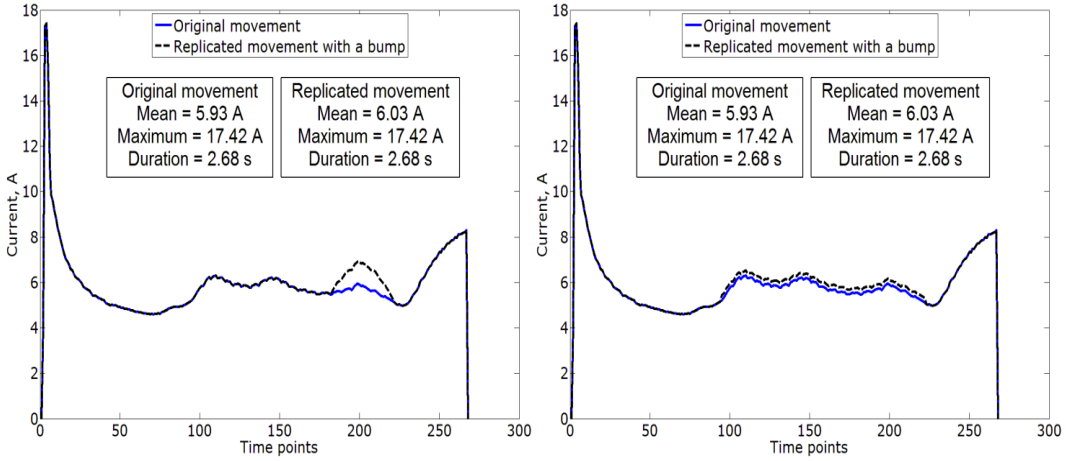


Figure 2.15. Different curves with the same average value

The difference between the two replicated curves can be observed visually. However with the alarm threshold technique both curves are treated as equal, since their average and maximum values of the measurements of current are identical as well as the motor running durations.

Another situation when the alarm threshold technique fails to identify the changes in the current shape, is when a peak and a valley in the shape of the curve cancel one another. The measurements of current of the same movement, as in previous example, are considered. This time a peak in the current is heightened, between time

points 135 and 159, and a valley is deepened by the same values, between time points 215 and 239. The resulting curves can be seen in Figure 2.16.

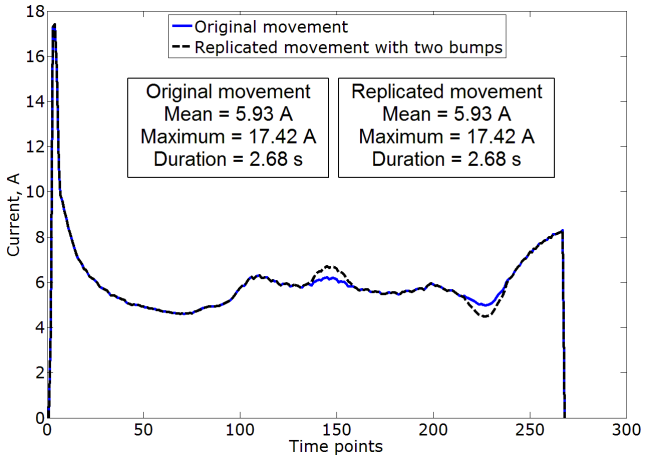


Figure 2.16. Different curves with the same average value

The alarm threshold technique treats both curves (original movement and replicated movement with two bumps) as equal, since their mean, maximum and duration values are identical. However, visually, a difference between these two curves can be observed.

These simple examples show that a lot of information can be lost if only the average and peak of the current is used. Even if both movements raised an alarm due to higher than normal average value, nothing more could be said about the performance of the system if visual inspection of the currents is not performed. The methodology that is developed in this study is capable of detecting that curves like the ones presented in these examples are different and provides a solution to overcome the main limitation of the alarm threshold technique.

2.2.5 Requirements for a new fault detection system

The main disadvantage of most of the fault detection systems described in section 2.2.2 is that they can detect a fault after it happens and cannot predict its occurrence. Only a few researchers so far tried to predict the deteriorating condition of railway point system (Yilboga et al., 2010, Eker et al., 2011, Chamroukhi et al., 2011). However, the systems, described in (Yilboga et al., 2010) (Eker et al., 2011), are based on modelling the degradation processes, thus their performance on real data still needs to be tested further. The approach described in (Chamroukhi et al., 2011) was tested on real data, however, the three tested classes were easily distinguishable and the approach needs further testing on the data with more subtle changes between the normal and defective operations. Taking into account features of existing fault detection systems and their deficiencies, requirements for a new fault detection and classification system (FDC) can be formulated:

- Able to identify a deteriorating condition prior to a fault occurrence;
- Universal in terms of using data from any type of point system;
- Able to deal with a lack of historical data of faulty conditions;
- Able to detect failure modes that were not present in historical data;
- Have a small as possible false alarm ratio;
- Have an automated and fast fault detection algorithm, suitable for on-line regime.

2.3 Possible approaches for the fault detection of railway point systems

As mentioned in section 2.1.6.3, the features of the data available in this study highly influenced the selection of a method for fault detection of railway point systems. The methodology proposed in this study is the first step towards achieving a fault detection and diagnostics system with the focus on fault detection. Considering the success of previously developed data-based approaches in the literature, the fault detection of railway point systems is considered as a classification task. Bayesian Belief Networks, K-Nearest neighbours, Neural Networks and Support Vector Machines are four commonly used classification methods. Each of them has its own advantages and disadvantages discussed next providing justification of the choice of the classification method used in this study.

2.3.1 Bayesian Belief Networks

The idea of a Bayesian Belief Network (BBN) is that certain features of the object influence the probability of the object belonging to distinct classes. Causal relationships between the features are defined and BBN becomes a probabilistic directed acyclic graph. The BBN approach is commonly used for fault detection when one considers a system consisting of different subsystems with sensors to measure parameters which describe the system state, e.g. water flow and level sensors used in a water tank system (Lampis and Andrews, 2009a). However, in this study only the current on a POE is measured and no other measurements are obtained, thus a BBN approach is not suitable for this particular problem. Furthermore, building a BBN usually requires some expert knowledge of system behaviour and prior probabilities of fault occurrence. Therefore, this model-based approach will not be used in this study.

2.3.2 K-nearest neighbours

K-nearest neighbours (K-NN) method was first introduced in an unpublished report by Fix and Hodges (Fix and Hodges, 1951) and it is one of the simplest classification algorithms. The idea of the K-NN is to find K neighbours in the training dataset for a new query object. Those K neighbours are assumed to be the closest to the query and the similarity is usually calculated by the Euclidean distance of the object features. The object is then classified based on the labels of K neighbours in the training dataset. When $K > 1$ the chosen label is the most frequent among the K neighbours, and when $K = 1$ it is the exact label of the neighbour. The classification might be problematic when using 2-NN, since there could be a chance that those 2 neighbours are labelled as belonging to different classes. However, in such a case to avoid any discrepancies the neighbour with a smaller distance can be chosen. The advantage of K-NN method is that it does not need any heuristics associated with the design of the model, as is the case with BNN and NN, since it uses all of the training data to find the K-nearest neighbours. However, this advantage might become a disadvantage when large datasets are used for the training of K-NN. All of the training data needs to be stored and used to classify the new object, thus the classifier performs slower, when the amount of the training data increases, which can cause problems when using it as an online fault detection approach. Thus, this classifier will not be used in this study.

2.3.3 Neural networks

The idea of Neural Network (NN) was first introduced by neurophysiologist Warren McCulloch and mathematician Walter Pitts (McCulloch and Pitts, 1943) and was later developed to a technique that is one of the most commonly used techniques to solve classification problems. The idea of NN is to input the data into a model, which consists of several interconnected layers of neurons, and get an output combined from the results of all neurons. Each neuron has an activation function, which transforms the input, and a defined weight for the output. Thus, using neurons with different weights and different activation function parameters allows NN to detect the non-linear relationships in the data and classify it accordingly. NN has been already applied for railway fault detection: for railway point systems fault prediction in (Yilboga et al., 2010) and railway track circuit fault detection in (Chen and Roberts, 2006, Chen et al., 2008).

However, the NN approach requires certain heuristics, as is the case with building BBN. The choice of the activation function of neurons and the number of neurons in each layer has to be made before training the model, however there is no clearly defined technique available on how to choose the optimal design of NN. The other disadvantage of NN is that it minimises the empirical risk of misclassification by selecting the parameters that allow NN to adapt to the training data very well. Using this approach a problem of over-fitting the model to the training data and losing generalisation might occur. This may lead to increased misclassification of new data. Thus, to avoid using heuristics in the design of the model and facing the over-fitting problems, the NN approach is not considered in this study.

2.3.4 Support vector machines

A fairly new technique, if compared to the other classification approaches e.g. NN 1942, K-NN 1951, was proposed by Boser, Guyon and Vapnik in 1992 (Boser et al., 1992) and later it was developed by Vapnik (Vapnik, 1995). The technique was named Support Vector Machine (SVM) and the idea of the technique was similar to NN. It is also a network as NN is, but here support vectors, instead of neurons, are combined. The support vectors are selected from the training dataset and thus the design of SVM is obtained from the training dataset. In addition to this difference, the training process of SVM is also different from the one in NN. SVM minimises the structural risk by balancing out the model complexity and its success to fit the training data, and finds the boundaries between different classes with the maximum margin of separation. In this way, fewer samples of training data are needed to obtain good results and the model can potentially be generic. Due to this structural risk minimisation, SVM overcomes the weakness of NN of over-fitting the model to the data. There are several variations of SVM and in this study One Class SVM (OCSVM) is chosen.

OCSVM (Scholkopf et al., 2001) is a special case of standard SVM, proposed for one-class classification problems, sometimes referred to as a novelty detection algorithm. This method has been chosen in this study due to the high imbalance of data available for two classes: good (fault-free) movements significantly outnumber faulty movements. There are several modifications of the traditional SVM algorithm introduced in literature to deal with imbalances in data, e.g. by using weights to penalise the class that is under-represented (Akbani et al., 2004, Imam et al., 2006). However, when there are only several failures of point systems per year, the data imbalance is enormous. For example, in total 589 movements were made in one direction by one point system considered in this study, from which 531 were fault-free

movements and 58 were faulty movements, giving a ratio of almost 1 to 10. Moreover, when some of the faulty movements are used in the training phase of the process, even fewer faulty movements are left for the testing phase. Thus, even using the weighted SVM approaches, too few faulty movements would be used in the analysis and thus OCSVM, which is trained on only one class data, has been chosen. More details about the OCSVM approach are given in section 2.3.5.

2.3.5 One class support vector machines

Scholkopf et al. suggested a new modification to SVM (Scholkopf et al., 2001) to make it suitable to be trained on one class data only. The idea is very similar to binary SVM, when two classes are used, but instead of trying to find the maximum margin between two classes, this method tries to separate the outliers from a given data set. These authors developed an algorithm that creates a function f, which has the value +1 for the most of the data points in a small region and the value -1 for the remaining points considered as outliers. Such outliers are separated from the most of the data points with the maximum margin.

If a training dataset is marked as $x_1, x_2, \dots, x_l \in X$, then this dataset can be mapped with a feature map $\Phi: X \to F$ into an inner product space F. The image of Φ can be calculated using a kernel, which is a key part of mapping the data into an inner product space. Different OCSVM kernel functions can be used: linear, polynomial, radial-basis function etc. A radial basis function (RBF) kernel is commonly used in a wide area of fault detection, such as fault diagnostics of power transformers (Ganyun et al., 2005), fault diagnostics of air conditioning systems (Li et al., 2010), fault detection of railway point systems (Asada and Roberts, 2011). The RBF kernel has also been used in this study, since it allows to choose a distance function used in the kernel (Dongyu et al., 2010):

$$k(x,y) = \big(\Phi(x) \cdot \Phi(y)\big) = \exp(-\gamma D^2(x,y),) \qquad \textbf{2.9}$$

where γ is the width parameter, $D(x,y)$ is the selected distance function, x and y are the feature vectors.

It is important to note that the distance function $D(x,y)$ is a similarity measure of two time series, if the time series are the only input parameter to OCSVM. This was the case in this study since measurements of current were the only input. Any function can be used as $D(x,y)$, as long as it is a metric. For the similarity measure to be a metric the following conditions must be met:

$$D(x,y) \geq 0, \qquad \textbf{2.10}$$

$$D(x,y) = 0 \text{ if and only if } x = y, \qquad \textbf{2.11}$$

$$D(x,y) = D(y,x), \qquad \textbf{2.12}$$

$$D(x,z) \leq D(x,y) + D(y,z), \qquad \textbf{2.13}$$

A default choice for this distance function is the Euclidean distance, sometimes referred to as L_2 norm (Deza and Deza, 2009). There are some similarity measures, e.g. Dynamic Time Warping (Sakoe and Chiba, 1978), that hold the first three conditions, but violate the last condition of triangle inequality. Such similarity measures cannot be used as the distance function in OCSVM. The Euclidean

distance with several other time series similarity measures that were used in this study will be presented in section 2.3.6.

The decision function of OCSVM is expressed as a hyper-plane:

$$f(x) = sgn\Big(\big(w \cdot \Phi(x)\big) - \rho\Big), \tag{2.14}$$

where w is the weight vector, ρ is the bias term of the hyperplane, $\Phi(x)$ is the feature mapping and sgn is a sign function, defined as $sgn(x) = \begin{cases} -1 \; if \; x < 0 \\ 1 \; if \; x \geq 0 \end{cases}$.

In order to find the decision function, a quadratic program is solved:

$$\min_{w \in F, \xi \in R^l, \rho \in R} \frac{1}{2}\|w\|^2 + \frac{1}{\nu l}\sum_i \xi_i - \rho \tag{2.15}$$

subject to

$$\big(w \cdot \Phi(x)\big) \geq \rho - \xi_i, \xi_i \geq 0, \tag{2.16}$$

where ν is a parameter that controls the number of support vectors and the fraction of the outliers, ξ are slack variables.

Using the Lagrangian multipliers $\alpha_i, \beta_i > 0$ the Lagrangian is formed as follows:

$$L(w, \xi, \rho, \alpha, \beta) = \frac{1}{2}\|w\|^2 + \frac{1}{\nu l}\sum_i \xi_i - \rho - \sum_i \alpha_i \Big(\big(w \cdot \Phi(x_i)\big) - \rho + \xi_i\Big) - \sum_i \beta_i \xi_i. \tag{2.17}$$

Setting the derivatives in respect to the variables w, ξ and ρ equal to zero, the variables w, ξ and ρ can be found from the following equations:

$$w = \sum_i \alpha_i \Phi(x_i), \tag{2.18}$$

$$\alpha_i = \frac{1}{\nu l} - \beta_i, \tag{2.19}$$

$$\sum_i \alpha_i = 1. \tag{2.20}$$

$\{x_i, \alpha_i > 0, i \in 1,2,\dots,l\}$ are called support vectors. Substituting the equations **2.18**, **2.19** and **2.20** into the equation **2.17**, the dual problem becomes:

$$\min_{\alpha} \frac{1}{2}\sum_{ij} \alpha_i \alpha_j k\big(x_i, x_j\big) \tag{2.21}$$

subject to

$$0 \leq \alpha_i \leq \frac{1}{\nu l}, \sum_i \alpha_i = 1\,. \tag{2.22}$$

When α_i values are determined the bias term ρ can be calculated in the following way:

$$\rho = w \cdot \Phi(x_i) = \sum_j \alpha_j k(x_j, x_i)\,, \qquad \textbf{2.23}$$

where x_i is a chosen support vector.

The decision function in the equation **2.14** can be rewritten using the equation **2.18** and the kernel expansion:

$$f(x) = sgn\left(\sum_i \alpha_i k(x, x_i) - \rho\right). \qquad \textbf{2.24}$$

The decision function, found by the algorithm described above, forms the optimal separating hyper-plane with the chosen values of parameters ν and γ. It is important to note that in order to get the optimal classification results, the optimal values of these parameters must be used as well. Therefore, a technique used for the determination of the optimal values of the parameters of OCSVM will be described in section 2.4.5.3.

In this study only the failure-free data is used in training. The training dataset should include a large range of good movements to represent the diversity detailed in section 2.1.6.3. Only then the OCSVM should be able to detect the faulty movements, which would be different from the ones in the training dataset. Another useful feature of the OCSVM is that a parameter, which controls the error allowed in the training data classification, can be set. This parameter bounds the fraction of the movements in the training dataset that are considered to be outliers, i.e. faulty movements. This parameter is usually a small number (e.g. 0.05), since it is expected that the majority of the data in the training dataset represents good movements, not faulty ones.

2.3.6 Time series similarity measures

As mentioned in the previous section, one of the key parts of the OCSVM method is the kernel function. With the RBF kernel, chosen in this study, there is a possibility to use different metrics instead of the Euclidean distance to define a similarity of two time series. To deal with the variable duration of the RPS operations, which is one of the main features of the real data described in section 2.1.6.3, distances or metrics that are able to measure the similarity of the time series of different lengths have to be used. Alternatively, rescaling of the data series should be done, since the standard Euclidean distance cannot be used for time series of different length. Moreover, the Euclidean distance compares two time series point by point and it does not take account of any variability in the data, e.g. peaks in the data might appear around the same time, but not at exactly the same time, thus the alignment given by the Euclidean distance might be misleading (an example is given in section 2.3.6.7).

Due to the growing interest in data mining techniques and time series matching methods to solve practical problems, a number of new similarity measures have been proposed, which allow elastic matching of time series, necessary in this study.

2.3.6.1 Elastic matching process

The idea of comparing two time series with elastic matching is based on the concept that was proposed by Levenshtein (Levenshtein, 1966) for the matching of alphabetical sequences. Three operations were used during the matching: insertion, deletion and reversion. Two sequences were compared to each other according to the number of edits needed for the sequences to become identical. For example, say two sequences “ABCDE” and “EBCDA” are compared. For these sequences to be identical, the first symbol “E” in the second sequence has to be replaced by symbol “A” and the last symbol “A” by symbol “E”. Thus the number of operations required is 2. This works well for alphabetical sequences, since it can be simply said, for example, letter “E” is not equal to letter “A”. However, when sequences of real values are compared, it cannot be said, for example, that number 4.5 is not equal to number 4.4, because the same thing can be said about the fact that 4.5 is not equal to 1000. The importance in this case is placed on how big the difference between the two numbers is and how this difference could be penalised.

Different variations on how to penalise for the edit operations have been derived for real-valued sequences, thus giving a wide variety of methods to choose from. Some of the methods commonly used in practice are: dynamic time warping (DTW) (Sakoe and Chiba, 1978), longest common subsequence (LCSS) (Das et al., 1997), edit distance with real penalties (ERP) (Chen and Ng, 2004), edit distance on real sequence (EDR) (Chen and Özsu, 2005) and time warp edit distance (TWED) (Marteau, 2009).

The output of the methods of the longest common subsequence and edit distance on real sequence is the total number of edits made on the sequences, as proposed by Levenshtein (Levenshtein, 1966). A threshold of two time series point difference is set in both methods to decide whether an edit operation is necessary. However, with these two approaches the information about the differences in the values of measurements is lost. For example, the differences in the measurements of current between two movements would be lost. Thus, these two approaches were not considered in the analysis.

A recently introduced technique called time warp edit distance differs from the other time series similarity measures, since it allows controlling the elastic matching of time series (Marteau, 2009). Such control of the elastic matching depends on a parameter, known as stiffness parameter γ. Another parameter is introduced to define the penalty for deleting an element from the time series, known as delete penalty λ. However, a procedure to find the optimal value of these two parameters can be computationally costly. The authors suggested using a cross-validation technique to find these optimal values. Furthermore, the optimised time warp edit distance performed similarly to the other state of the art similarity measures when testing was performed on selected data sets from a widely used time series classification data repository of University of California, Riverside (Keogh et al., 2011). The proposed approach of optimised TWED achieved the best result in 10 of 20 tested datasets and thus might be performing differently based on the data used. Due to these features, this time series similarity measure was also not considered in this study.

Dynamic time warping and edit distance with real penalties techniques were chosen for the measuring of time series similarity in this study. Their performance, in terms of fault detection accuracy, will be compared with the traditional time series similarity measure – Euclidean distance. The chosen techniques and the proposed approaches of their combination are presented in the following section.

2.3.6.2 Euclidean distance

A time series similarity is usually described using a distance function. For the two time series of equal length, the Euclidean distance is commonly used as a similarity measure. If two sequences of real valued numbers $A = (a_1, a_2, \dots, a_N)$ and $B = (b_1, b_2, \dots, b_N)$ are considered, then the Euclidean distance can be calculated as follows:

$$\delta_E = \sqrt{\sum_{i=1}^{N} (a_i - b_i)^2}, \qquad \textbf{2.25}$$

where δ_E is the Euclidean distance, a_i and b_i are the i-th elements of sequences A and B respectively, N is the length of sequences A and B.

The advantage of the Euclidean distance is that it is very fast to compute (takes $O(n)$ operations). However, it is defined only for equal length time series. Furthermore, it performs one to one matching and is not able to cope with some variations in x-axis, e.g. a slightly different start time of the locking phase of the point system that is often observed in the in-field data. Such an incapability to deal with shifts of the time series might lead to errors in similarity measures, even when the overall shape of the two time series is similar. A simple example on how the Euclidean distance produces misleading alignments, when two time series are very similar, but one is slightly shifted in x-axis, is given in section 2.3.6.7.

2.3.6.3 Dynamic time warping (DTW)

A novel approach for comparing two time series, called Dynamic Time Warping (DTW), was first introduced in the field of speech recognition (Sakoe and Chiba, 1978). Since the data for speech recognition greatly depends on the speed of the speaker and their tone, even the same words or letters will have observations of different length if sampled at the same sampling rate. To deal with these differences in the time scale and to match two similar words, a new method based on dynamic programming was introduced.

Consider two words that have been pre-processed into a sequence of feature vectors: $A = (a_1, a_2, \dots, a_N)$ and $B = (b_1, b_2, \dots, b_M)$, where $N \neq M$. Then an $N \times M$ matrix, DTW, consisting of differences between feature vectors A, B can be constructed: $DTW(i,j) = \delta_E(a_i, b_j),\ i = 1,2,\dots,N, j = 1,2,\dots,M$, where δ_E is usually the Euclidean distance. The elements of the new matrix correspond to the alignment between two points a_i and b_j. From the differences matrix we can find an optimal warping path $W = w_1, w_2, \dots, w_k, \dots, w_K$, $w_k = (i(k), j(k))$, $\max(N,M) \leq K < N + M - 1$, using dynamic programming. There are certain constraints for a path to be a warping path:

- **Monotonic conditions.** Points in the warping path W must be monotonically spaced in time, i.e. $i(k-1) \leq i(k)$ and $j(k-1) \leq j(k)$.
- **Continuity conditions.** This condition restricts the warping path W to go to adjacent cells only, i.e. $i(k) - i(k-1) \leq 1$ and $j(k) - j(k-1) \leq 1$.
- **Boundary conditions.** The warping path must start from the first elements a_1 and b_1 in the sequences A and B and finish with the last elements a_N and b_M, i.e. $i(1) = 1, j(1) = 1$ and $i(K) = N, j(K) = M$.

- **Adjustment window condition.** In some cases, to speed up the algorithm or to avoid improper alignments between elements that are too far away, a windowing condition is used: $|i(k) - j(k)| \leq r$, where r is the window size. To acquire the original version of dynamic time warping method without the window, the window size must be set as $r = \max(N, M)$.

A lot of paths in the DTW matrix satisfy the above conditions and the optimal path that minimises the warping cost can be found from the following expression:

$$W = \min \sqrt{\sum_{k=1}^{K} w_k}. \qquad \textbf{2.26}$$

This path is found using dynamic programming to evaluate the recursion of the following cumulative distance:

$$\gamma(i,j) = \delta_{\mathrm{LP}}(a_i, b_j) + \min \begin{pmatrix} \gamma(i-1, j-1), \\ \gamma(i-1, j), \\ \gamma(i, j-1) \end{pmatrix}, \qquad \textbf{2.27}$$

where δ_{LP} is a selected L^P-norm, usually L^2 norm, also known as the Euclidean distance.

Then the dynamic time warping distance is given by:

$$\delta_{DTW}(A, B) = \gamma(N, M). \qquad \textbf{2.28}$$

One of the weaknesses of classical DTW is that the method does not take into account the shape of the time series. For example, if two identical data points are considered and one is on a rising trend and the other one is on a falling trend, classical DTW gives a zero difference between these data points, which might be misleading. To deal with this problem, Keogh and Pazzani introduced a modification to classical DTW, called derivative dynamic time warping (DDTW) (Keogh and Pazzani, 2001). They showed that if one would slightly change a local feature in a time series, e.g. the depth of a valley or the height of a peak, classical DTW would try to explain the difference by the distortion of X-axis. The reason for this is that DTW considers the values of Y-axis, and not the gradient of the curve.

To overcome this weakness, Keogh and Pazzani improved the algorithm by replacing the values of time series in Y-axis with the estimates of the first derivative of these values. The original DTW method can be used without any changes to the algorithm itself, only in this case the estimates of derivatives are the input, instead of values themselves. For noisy datasets, the authors used the exponential smoothing approach to reduce the noise before estimating the derivatives. The idea to use the estimates of the derivatives to compare two time series was also adapted to other time series similarity measures used in this study. The use of estimates of the derivatives will be described as one of the pre-processing techniques in section 2.4.3.

The major drawback for DTW, however, is that it is not a metric and it does not satisfy the triangle inequality given by **2.13**. The simple way to explain the violation of the triangle inequality is by saying that A is similar to B and B is similar to C, but A is not similar to C. For this reason, DTW cannot be used as a replacement of the Euclidean distance in the kernel function of OCSVM. A different approach how to use DTW for time series similarity is proposed in section 2.3.8.

2.3.6.4 Edit distance with real penalties (ERP)

Chen and Ng proposed an alternative method, similar to DTW, which can support a local time shifting, but is a metric at the same time, a necessary requirement to be used in the kernel function in OCSVM (Chen and Ng, 2004). The idea of ERP, was based on combining the L1-norm (or absolute value) and the distance function used for the comparison of strings, known as string edit distance:

$$\delta_{STR}(s_1, s_2) = \begin{cases} 0, & if\ s_1 = s_2 \\ 1, & if\ s_1\ or\ s_2\ is\ a\ gap, \\ 1, & otherwise \end{cases} \quad \textbf{2.29}$$

where s_1 and s_2 are some symbols.

The authors transformed the distance in **2.29** to be suitable for comparing sequences of real number values:

$$d_{ERP}(x, y) = \begin{cases} |x - y|, & if\ x\ and\ y\ are\ not\ gaps, \\ |x - g|, & if\ y\ is\ a\ gap, \\ |y - g|, & if\ x\ is\ a\ gap, \end{cases} \quad \textbf{2.30}$$

where x and y are real numbers, g is a gap penalty.

Consider the two time series $A = (a_1, a_2, \dots, a_N)$ and $B = (b_1, b_2, \dots, b_M)$, where $N \neq M$. Before calculating the ERP distance, normalisation of the time series needs to be done as indicated by the authors. The norming procedure will be described in a section about the pre-processing techniques 2.4.3.2. Denote the subsequence of A as $A_i^p = (a_i, a_{i+1}, \dots, a_p)$ and the subsequence of B as $B_i^q = (b_i, b_{i+1}, \dots, b_q)$. Then the ERP distance can be defined as follows:

$$\delta_{ERP}(A, B) = \begin{cases} \sum_{i=1}^{N} d_{ERP}(a_i, gap), & if\ M = 0, \\ \sum_{i=1}^{M} d_{ERP}(b_i, gap), & if\ N = 0, \\ \min \begin{Bmatrix} \delta_{ERP}(A_1^{N-1}, B_1^M) + d_{ERP}(a_N, gap), \\ \delta_{ERP}(A_1^{N-1}, B_1^{M-1}) + d_{ERP}(a_N, b_N), \\ \delta_{ERP}(A_1^N, B_1^{M-1}) + d_{ERP}(b_M, gap) \end{Bmatrix}, & otherwise. \end{cases} \quad \textbf{2.31}$$

Thus this metric uses a real penalty between two non-gap elements and a constant penalty value for the gap. The authors proved that for any fixed constant g the triangle inequality **2.13** holds, thus they suggest using $g = 0$ for two reasons. The first reason is that, when $g = 0$, the distance between two time series A and B corresponds to the difference between the area under A and under B, if they are plotted with x-axis representing the time points and the y-axis representing the values of time series. The second reason is that, to match A and B, the shorter time series is aligned to the longer one using gap elements. Since the value of the gap element is equal to a gap penalty g, which in this case is zero, no changes to the area under the extended time series are made.

2.3.6.5 ERP combined with Euclidean distance

Since time series are normalised before the ERP distance is calculated, the information about their difference in absolute values is lost. Therefore, in this study a new idea of how to keep the information about both differences in absolute values and in shapes is proposed. For this purpose, the ERP was combined with the Euclidean distance in the following way: the alignment of the two time series is calculated on the normalised time series using ERP and then the Euclidean distance of non-normalised time series is calculated using the alignment given by ERP. The newly proposed combination of two metrics, denoted δ_{EE}, is the result of ERP and the Euclidean distances combined in the following way:

$$\delta_{EE}(a,b) = 0.5 \cdot \delta_{ERP}(a,b) + 0.5 \cdot \delta_E(\hat{a},\hat{b}), \qquad \textbf{2.32}$$

where a and b are the original time series, $\delta_{ERP}(a,b)$ is the result of the ERP method, $\hat{a}$ and $\hat{b}$ are the time series after they are warped with the ERP method, $\delta_E(\hat{a},\hat{b})$ is the Euclidean distance of the warped time series $\hat{a}$ and $\hat{b}$. The weights are chosen 0.5 for both distances, so that the same penalty is given for the difference in the shape of the time series and the difference in the absolute values, but other weight could be chosen if it is necessary to put more emphasis on the similarity based on either shape or absolute values of the time series. The results of the application of combined metrics to evaluate for the time series similarity measures were compared to the application of separate metrics and will be presented in section 2.5.

2.3.6.6 DTW combined with Euclidean distance

A similar approach can be proposed for combining the DTW distance and the Euclidean distance. The alignment of the two time series is calculated using DTW and then the Euclidean distance of non-normalised time series is calculated using the alignment given by DTW. The result of DTW and the Euclidean distance is then combined in the following way:

$$\delta_{DE}(a,b) = 0.5 \cdot \delta_{DTW}(a,b) + 0.5 \cdot \delta_E(\hat{a},\hat{b}), \qquad \textbf{2.33}$$

where a and b are the original time series, $\delta_{DTW}(a,b)$ is the result of DTW method, $\hat{a}$ and $\hat{b}$ are the time series after they are warped with DTW method, $\delta_E(\hat{a},\hat{b})$ is the Euclidean distance of the warped time series $\hat{a}$ and $\hat{b}$. The weights are chosen 0.5 for both distances, so that the same penalty would be given for the difference of the shape of the time series and the difference of the absolute values. However, as in the case of DTW, this similarity cannot be used as a replacement of the Euclidean distance in the kernel function of OCSVM. It was used as a feature extraction technique as it will be explained in section 2.3.8.

2.3.6.7 Illustrative comparison of Euclidean distance and elastic similarity measures

A simple example is considered in this section to demonstrate and compare the capability of different similarity measures to compare two time series, which has x-axis scaling. For illustration purposes, a part of measurements of current of one railway point system is taken and a shift along x-axis is added to obtain a replicated time series. The shift was added from the 15th point of time series by replicating the same value of current several times. The replicated time series is trimmed by the same amount of elements as it was lengthened, since the Euclidean distance can

only be applied on the same length data series. This way, the comparison between the Euclidean distance and the elastic similarity measures will be performed on the identical data. The original and replicated time series can be seen in Figure 2.17.

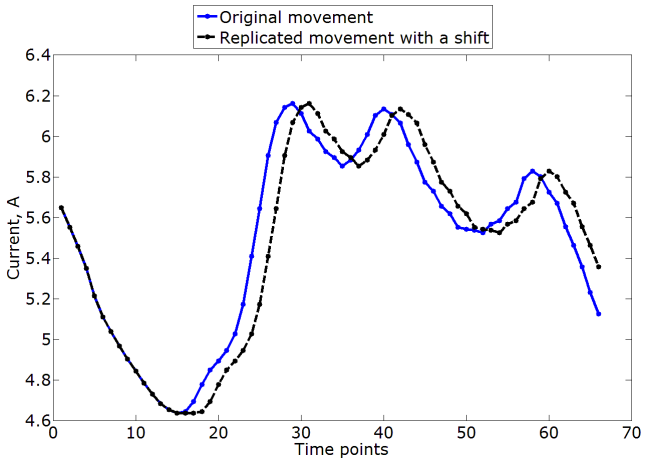


Figure 2.17. Artificial problem for the time series alignment

The alignments, calculated by the Euclidean distance, DTW and ERP methods, are given. Each alignment is visually compared with the perfect alignment, which should be obtained considering the added shift in the time series. To visualise the alignment, the replicated movement was shifted along the y-axis in the plot. The alignment of two time series is given by lines between two points:

- Red line indicates that the alignment of two points is not perfect.
- Green line indicates that the alignment of two points is perfect.

Firstly, the Euclidean distance is used for the alignment of the two time series. The resulting alignment is plotted in Figure 2.18 on the left hand side and the perfect alignment is plotted on the right hand side.

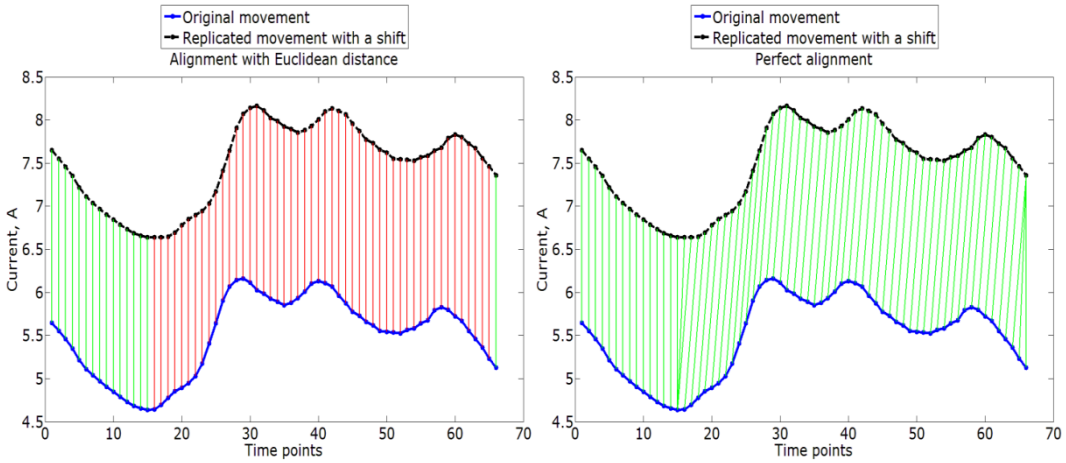


Figure 2.18. Alignment by Euclidean distance (left figure) and perfect alignment (right figure)

As explained before, the Euclidean distance method aligns the two time series by one to one matching and thus it is incapable of considering any scaling of the x-axis. When the same peaks or valleys appear in the two time series, but they are slightly shifted in the x-axis, which is commonly observed in the in-field data of railway point systems, the similarity of the two curves given by the Euclidean distance might be inaccurate, since it ignores this shifting.

The alignment given by the DTW similarity measure is plotted in Figure 2.19, on the left hand side and the perfect alignment is plotted on the right hand side.

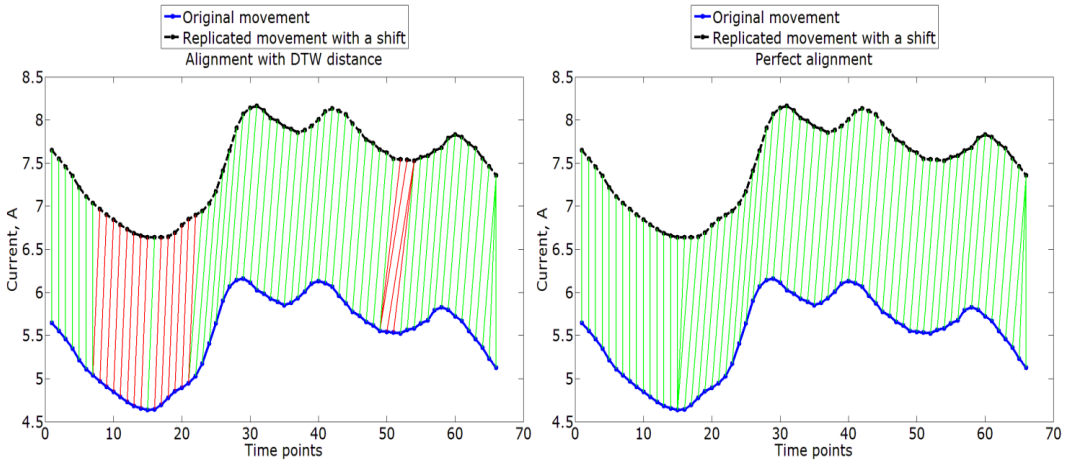


Figure 2.19. Alignment by DTW (left figure) and perfect alignment (right figure)

The DTW method is able to deal with the shifting of the time series, since most of the peaks were aligned after the shifting. However, it produced some unnecessary warping around time point 50. This is undesirable, since the resulting warping distorts the result of the similarity measure of the two time series by adding penalty for the matching of irrelative data points, when no matching of these points was needed and no penalty should be given. Moreover, it has some difficulties when dealing with the parts of the time series which are flat. This however is quite expected, since the values of the time series in the flat parts are of small difference and thus the elastic matching is penalised with a small penalty only.

The alignment given by the ERP metric is plotted on the left hand side and the perfect alignment on the right hand side of Figure 2.20

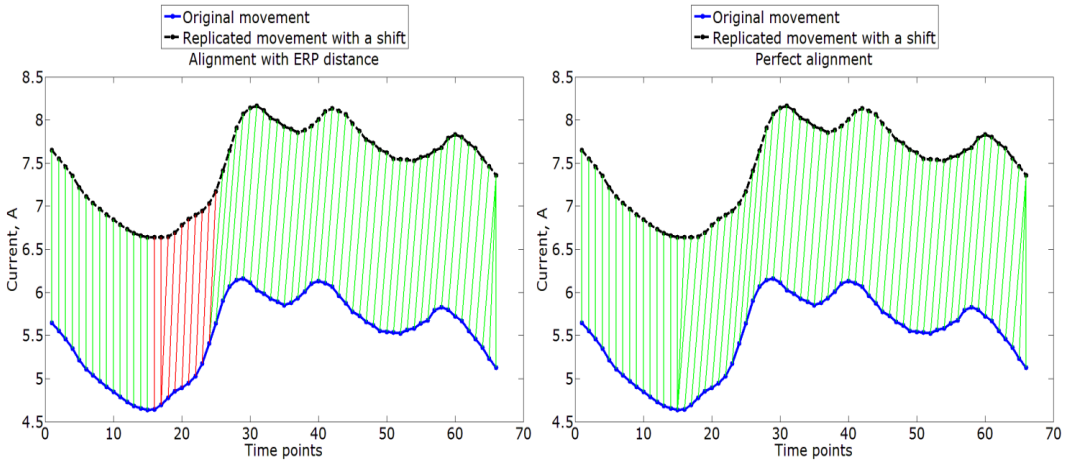


Figure 2.20. Alignment by ERP distance (left figure) and perfect alignment (right figure)

It can be seen that the alignment given by the ERP distance is the closest to the perfect alignment. ERP was able to deal with the shifting of the time series and did not produce unnecessary warping, as in the case of DTW. However, the alignment given by the ERP deviated from the perfect alignment in the time points where the shift was added to the original movement. Thus, several points were aligned inadequately until the ERP method identified the shift in the time series.

It can be concluded that the two chosen elastic similarity measures allow a better alignment of the time series with shifts than the Euclidean distance. However, sometimes they can produce unnecessary warping of the x-axis, as observed in the case of DTW.

2.3.7 An approach of using time series similarity measure as a kernel distance function of OCSVM

Any time series similarity measure that is also a metric can be used within OCSVM, replacing the traditional Euclidean distance in the OCSVM kernel, as explained in section 2.3.5 and suggested in (Dongyu et al., 2010). Thus ERP distance is the only suitable option from the similarity measures in section 2.3.6. Thus in the case of the ERP metric, the kernel function of OCSVM becomes:

$$k(x, y) = \exp\left(-\gamma \delta_{ERP}^2(x, y)\right), \qquad \textbf{2.34}$$

where x and y are the time series of the measurements of current, δ_{ERP} is the ERP distance as defined by equation **2.31**, γ is the OCSVM parameter as described in section 2.3.5.

Using a similarity measure as a kernel distance function requires all of the movements to be compared one by one. For example, if 200 movements were used for the analysis, then 40000 (200^2) comparisons of similarity measures should be made. This number of comparisons can be decreased by exploiting several features of the metrics, given by equations **2.11** and **2.12**. Therefore, the number of comparisons can be decreased more than twice, i.e. from 40000 to 19900, but still a large number of comparisons for 200 movements need to be carried out. The number of comparisons, N_C, needed for any number of movements, N_T, can be calculated with the following formula:

$$N_C = \frac{N_T!}{2 \cdot (N_T - 2)!} = \frac{N_T \cdot (N_T - 1)}{2} \qquad \textbf{2.35}$$

Thus, the time taken to train the OCSVM model increases greatly when the number of movements used increases.

2.3.8 An approach of using time series similarity measure as a feature input to OCSVM algorithm

To avoid large number of comparisons of movements made in OCSVM as discussed in 2.3.7, similarity measures can be utilised to compare the data of several previous movements prior to applying the OCSVM algorithm. Then the result of the similarity measure is used as an input time series x and y in the default OCSVM kernel defined by equation **2.9**. This approach does not limit the choice of the similarity measure, thus non-metric similarity measures, such as DTW, can also be used. Using the similarity measure to calculate the similarity of the chosen movement with several previous movements, the current movement could be compared with one, two, five or ten previous movements. Thus the feature vectors of OCSVM would be of length 1, 2, 5 and 10 respectively.

This approach requires fewer comparisons of movements than the first one presented in section 2.3.7. If considering the above described example, when 200 movements are used in the analysis, with this approach the number of comparisons

to be made depends on how many previous movements are used to calculate the similarity. For example, if 10 previous movements are considered, then 2000 comparisons should be made. The number of comparisons needed, N_C, can be evaluated in the following way:

$$N_C = N_T \cdot N_P, \quad \textbf{2.36}$$

where N_T is the number of movements used and N_P is the number of the previous movements considered.

2.3.9 Summary

A variety of classification methods, such as Bayesian Belief Networks, K-nearest neighbours, Neural Networks and Support Vector Machines were considered for the fault detection of railway point systems in this section. A brief introduction to each method was given, stating whether it could be used for the fault detection of in-field railway point systems. A modification of SVM, known as OCSVM, was chosen in this study, mainly due to the fact that it could deal with the features observed in the data of in-field railway point systems.

After determining the most suitable fault detection method, the second key task in this study was to choose a method to compare the data obtained from the in-field railway point systems whose results would then be used in a fault detection model (OCSVM). Since the measurements of current were the only observed parameters in this study and for each operation of the railway point system the measurements of current in amperes over time formed a time series, time series comparison algorithms were considered.

Various similarity measures to compare two time series were described in this section. The default similarity measure to compare two time series is the Euclidean distance. Alternative similarity measures, known for the elastic matching they perform, were presented. For the data available in this study, they have a possibility to overcome the weaknesses of the Euclidean distance. One such weakness of the Euclidean distance was the inability to deal with the shifts in time axis of the measurements of current. The introduced elastic similarity measures were designed to solve such problems. Further, a combination of Euclidean distance and elastic similarity measures was proposed in this study. The combination of these two similarity measures was formed to take into account the shape of two time series, while keeping the information about the differences in the absolute values. The comparison of the results, when the similarity measures were used separately to the ones when the combined similarity measures were used, will be presented in section 2.5.

Two approaches on how the time series similarity measures can be used in conjunction with the OCSVM method were proposed. One of the suggested approaches was to use the similarity measure as the RBF kernel distance function of OCSVM. In the other approach, the similarity measure was used to pre-calculate the similarity between several previous movements, before using these as feature vectors of the OCSVM. Both approaches were tested in this study and the results are presented in section 2.5.

2.4 Proposed methodology for the fault detection of railway point systems

2.4.1 Introduction

The first step of the proposed methodology is the automatic labelling of data according to the chosen labelling option (alarm or fault dates), as presented in section 2.1.6.4. Since any two measurements of current of the same railway point system differ in length, as presented in section 2.1.6.3.1, the standard Euclidean distance (section 2.3.6.2) cannot be used without modifying the data first (section 2.4.3.1). An alternative way of comparing two different length time series is to use the elastic metrics on original data. The choice of the elastic metric, such as ERP, is justified in section 2.3.7.

The effectiveness of elastic metrics and Euclidean distance will be evaluated considering fault detection accuracy (section 2.4.5). After the choice of the similarity measure, various pre-processing techniques can be used in order to speed up the calculation of the similarity measure or in some cases provide a better matching of the two time series (section 2.4.3). After the data was pre-processed (or no pre-processing performed) the pairwise similarity measures for all movements are calculated. The fault detection on railway point systems in this study is considered as a classification task. The features observed in the data of in-field RPS (section 2.1.6.3) had a great impact on choosing an appropriate method used for the fault detection problem. Due to a high imbalance between the data in each class, traditional multi-class or binary supervised learning classifiers are not suitable, as discussed in section 2.3. The one-class classifiers should be used instead, which are trained only on the data of good movements to detect abnormal movements. The OCSVM method (section 2.3.5) is chosen in this study.

The similarity measures with the OCSVM model can be used in two ways, as described in sections 2.3.7 and 2.3.8. The next step is the choice of OCSVM training and testing technique. Two options will be presented in sections 2.4.5.1 and 2.4.5.2. A grid of OCSVM parameter values is selected next in order to find the best combination of parameter values which leads to the highest movement classification accuracy. Each combination of OCSVM parameter values is used for training and testing the model and the one with the highest classification accuracy is selected as will be described in section 2.4.5.3. The steps of the methodology can be summarised in the following flowchart in Figure 2.21.

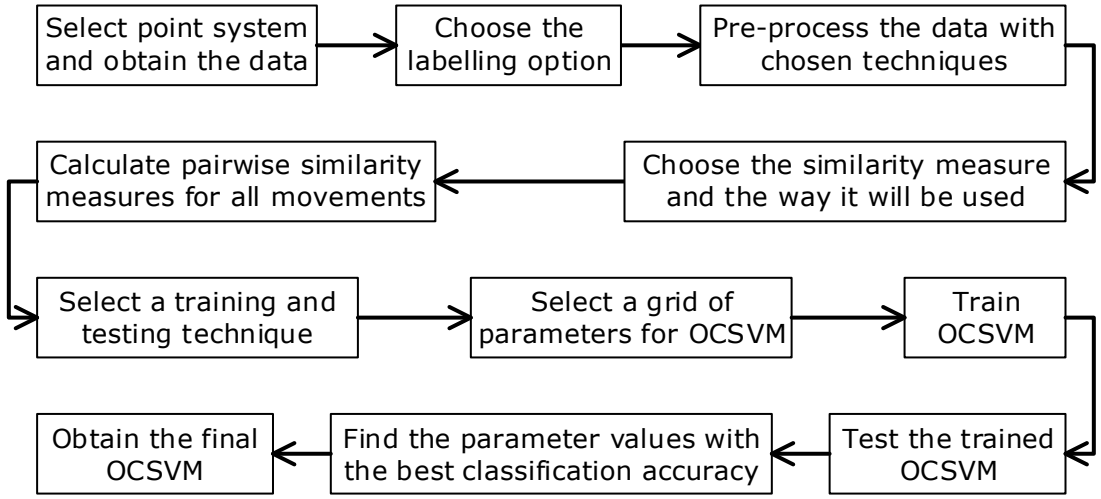


Figure 2.21. Flowchart of proposed methodology

2.4.2 Data input

Data for the analysis is stored in Matlab structure arrays. This way, various data types (dates, symbolic data, double, integer values) can be kept together in one place. The Matlab structure consists of fields described in Table 2.2.

Table 2.2. Fields of Matlab structure

Field name	Field type	Description
PM_ID	Character	POE ID
Direction	Double	Direction of the POE operation (1 – normal to reverse, 2 – reverse to normal)
Date	Date string	Date and time of the start of operation of POE
Current	Vector of doubles	Current measurements (in amperes) of the POE pump motor
Time	Vector of doubles	Time points when the current was measured
Duration	Double	Duration of POE operation
Target	Double	Class number indicating fault state (-1) or fault-free state (1)

Depending on how a chosen similarity measure is used, two types of OCSVM input data exist. If the similarity measure is used as a kernel function, then the time series of measurements of current are the inputs to the model, denoted as $\boldsymbol{x} = (Current_1, Current_2, \dots, Current_N)$, where N is the total number of movements. If the similarity measure is used to compare the similarity of several previous movements, then the input to the model will be the results of similarity measures for previous movements $\boldsymbol{x} = (\boldsymbol{x_1}, \boldsymbol{x_2}, \dots, \boldsymbol{x_N})$, where N is the total number of movements,$\boldsymbol{x_i} = (f_1\ f_2 \dots f_M)$ and f_i is the value of similarity measure between the given movement and the ith previous movement and M is the number of previous movements considered. The label of each ith movement will be entered into the model as $L_i^t = -1$ for faulty movements, and $L_i^t = 1$ for good movements. This information will be stored in vector $\boldsymbol{L^t}$, t in the superscript meaning that it is the true label of movement, to distinguish it from the label given by the output of OCSVM. Two automatic labelling approaches, as described in section 2.1.6.4, are considered, one based on the alarm levels and the other based on fault dates.

2.4.3 Data pre-processing

In the data pre-processing step, first of all, any trailing zeros after the completion of a movement were removed from each current time series. Secondly, rescaling of time series with uniform scaling (Fu et al., 2008) was done in order to be able to use the Euclidean distance as a similarity measure and compare the results with elastic metrics on identical data. Finally, the first 30 milliseconds of each movement were removed from the time series. This phase of movement is known as motor inrush and thus, as recommended by engineers, can be left out of the analysis.

Several other data pre-processing techniques have been proposed to speed up calculations and/or possibly give better data alignments when using different similarity measures:

- Smoothing of the time series with the exponential smoothing with different smoothing factor, α, values to filter out noise (Keogh and Pazzani, 2001).
- Down sampling of the time series by an integer factor to speed up the calculations (Vuori et al., 2001).
- Norming of the time series (Chen and Ng, 2004).

- Calculating the estimates of the first derivatives for each data point in the time series (Keogh and Pazzani, 2001).

The flowchart of possible options of the data pre-processing process is given in Figure 2.22.

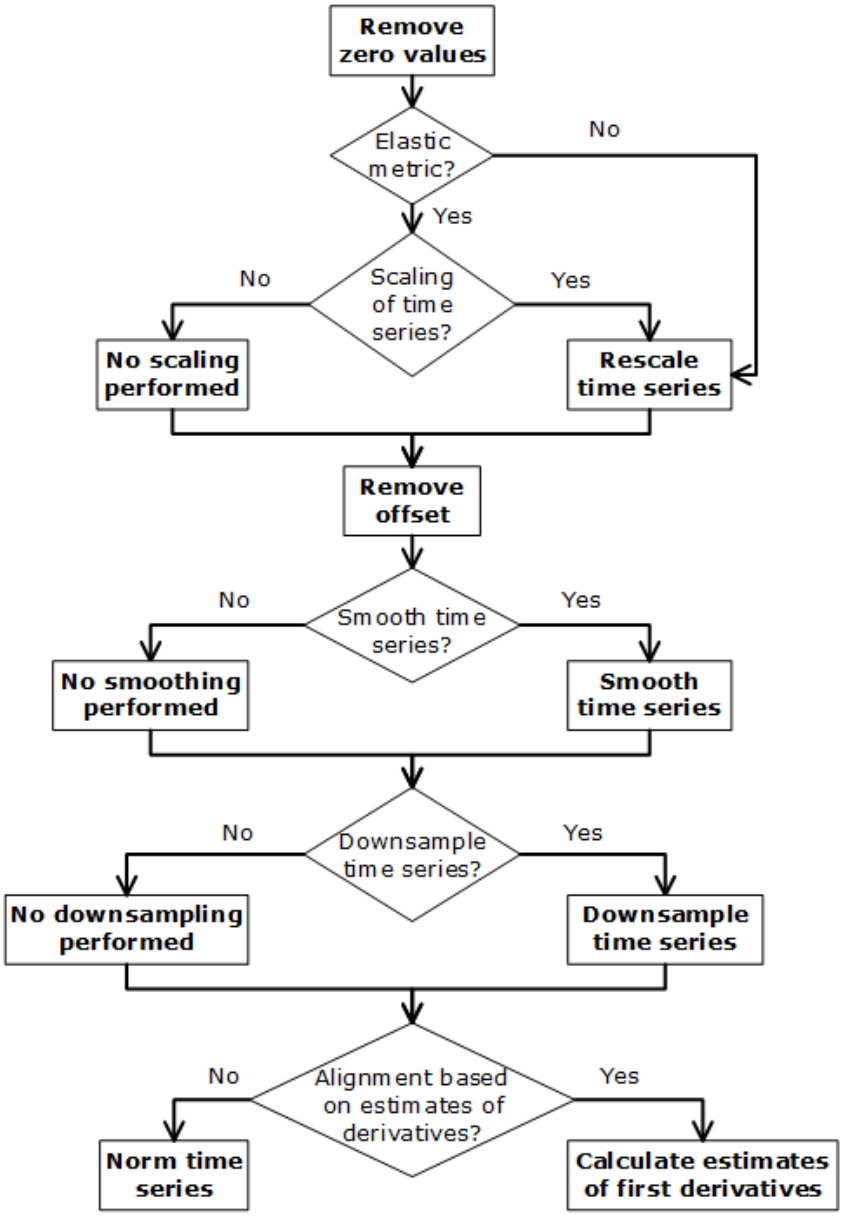


Figure 2.22. Flowchart of data pre-processing

Different options of data pre-processing have been implemented individually and collectively. Procedures for implementing these options and the expected gain of their application are discussed in detail in further subsections.

2.4.3.1 Rescaling time series

In order to compare OCSVM performance results obtained using elastic metrics and the Euclidean distance, the time series have to be rescaled to the same length, as discussed in section 2.1.6.3. The rescaling is done by uniformly scaling the time

series to an average length of all movements of the same point system used in the analysis, according to a formula (Fu et al., 2008):

$$a_i^R = a_{\lceil i \cdot N/M \rceil}, i = 1,2, \dots, M, \quad \textbf{2.37}$$

where a_i^R is the i-th element of the rescaled time series, N is the length of the original time series, M is the length of the rescaled time series, $\lceil i \rceil$ is the ceiling function of i which calculates the smallest integer not less than i, $a_{\lceil i \cdot N/M \rceil}$ is the $\left\lceil i \cdot \frac{N}{M} \right\rceil$ element of original time series.

2.4.3.2 Norming time series

The norming of time series is carried out before aligning them for similarity comparison, as proposed by the authors of the ERP distance (Chen and Ng, 2004). For the time series $A = (a_1, a_2, \dots, a_N)$ the norming is done in the following way:

$$a_{norm}(i) = \frac{a(i) - \mu(A)}{\sigma(A)}, i = 1, \dots, N, \quad \textbf{2.38}$$

where $a(i)$ is the i-th element of the time series A, $\mu(A)$ is the mean value of the time series A, $\sigma(A)$ is the standard deviation of the time series A, and N is the length of time series A.

The norming of time series removes the offset between the time series and allows a better matching. However, the information about the absolute values of two time series is lost, and as for some failure modes (e.g. dry slide chairs, obstruction) an increase of the current usage is experienced, these failure modes might be left undetected, if absolute values are ignored. A simple example can be considered, when the norming of the time series leads to a false decision about the time series similarity. A shift across the y-axis by 1 Ampere is added to obtain the replicated time series. The original and replicated time series can be seen in Figure 2.23.

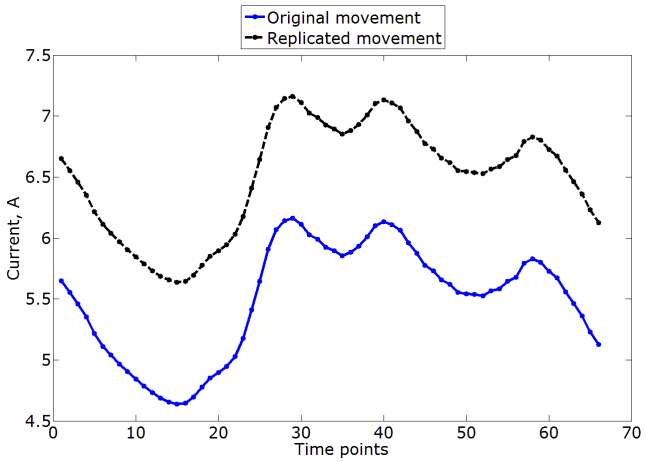


Figure 2.23. Artificial problem to illustrate the loss of information after norming the time series

After norming the time series, the alignment given by ERP is the same as perfect (all lines connecting the points of two time series are green), as can be seen in Figure 2.24.

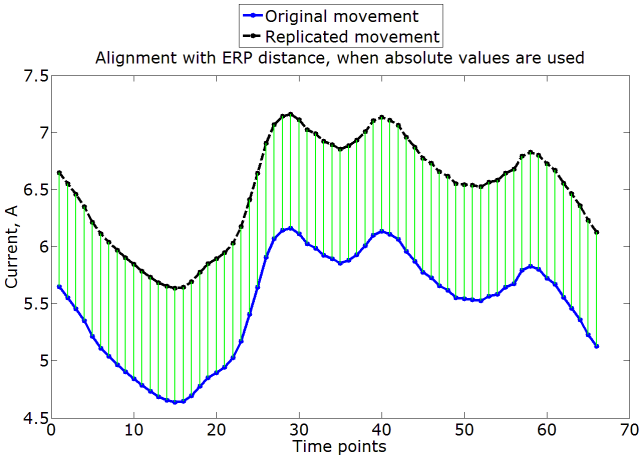


Figure 2.24. Alignment of two time series with ERP distance

These two time series were found to be identical, since the result of the ERP distance was zero. However, these two time series only have identical shapes, but the difference between their absolute values is obvious. Thus, norming of the time series might result in a loss of detection of certain faults with excess of current used.

This problem should be addressed by using Euclidean distance combined with elastic metrics, as proposed in sections 2.3.6.5 and 2.3.6.6.

2.4.3.3 Exponential smoothing

Some level of noise exists in any type of measurements of electric equipment using sensors. There are various methods to de-noise the time series, e.g. moving average filter (Lorenz and Köhler, 2005), exponential smoothing (Lorenz and Köhler, 2005), Kalman filter (Kalman, 1960). To smooth the time series obtained from the measurements of POE current, the exponential smoothing was used in this study due to its simplicity. For the time series $A = (a_1, a_2, \dots, a_N)$, the exponential smoothing is defined as follows:

$$a_j^S = a_{j-1}^S + \alpha\left(a_j - a_{j-1}^S\right), j = 2,3, \dots, N,\ a_1^S = a_1, 0 < \alpha < 1, \qquad \textbf{2.39}$$

where a_j is the *j*th data point of the original time series, a_j^S is the *j*th data point of the smoothed time series, α is the smoothing factor.

The removal of noise is expected to improve the alignment of time series based on estimates of the derivatives of their values as mentioned in section 2.3.6.3. The positive effect of the exponential smoothing is shown in an example when two time series, with and without smoothing, are matched. In both cases, the estimates of the derivatives are used as an input to ERP. The time series without smoothing are plotted in Figure 2.25. As in the previous example in Figure 2.17 the original time series was shifted and additionally two bumps were added to test the matching produced by the ERP distance.

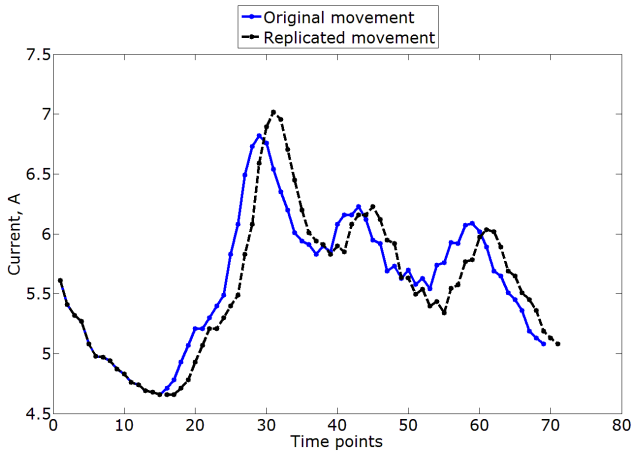


Figure 2.25.Time series without smoothing

After smoothing the original time series, the smoothed time series are given in Figure 2.26.

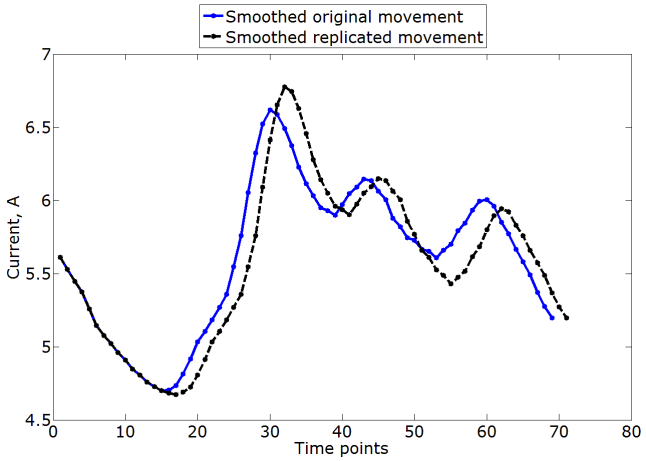


Figure 2.26.Time series after exponential smoothing

Since the replicated time series was created artificially, the perfect alignment of the time series is known. Thus, the alignment produced by ERP on smoothed and non-smoothed data is then compared with a perfect alignment that should be produced given these two time series. As in the previous example, the alignment that is deviated from the perfect is plotted with red lines. First, the alignment on the non-smoothed data is given in Figure 2.27.

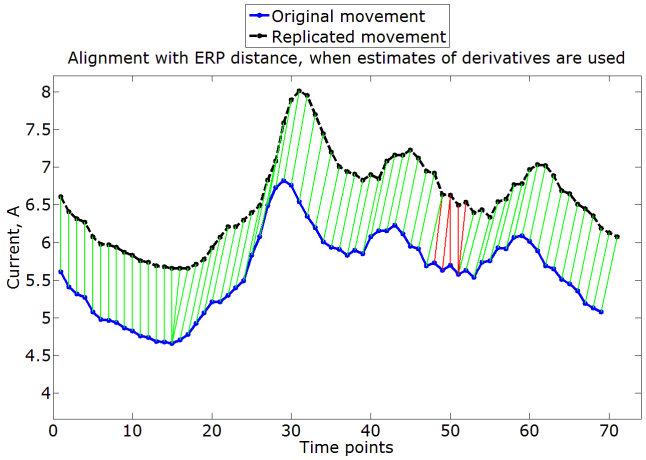


Figure 2.27. ERP alignment on non-smoothed data

Most of the data was perfectly aligned, except for the data points around time point 50. A presence of noise can be visually observed in that part of the time series. This might be the cause of why the ERP failed to align those points perfectly. The alignment produced by ERP, when the data was smoothed with exponential smoothing is plotted in Figure 2.28.

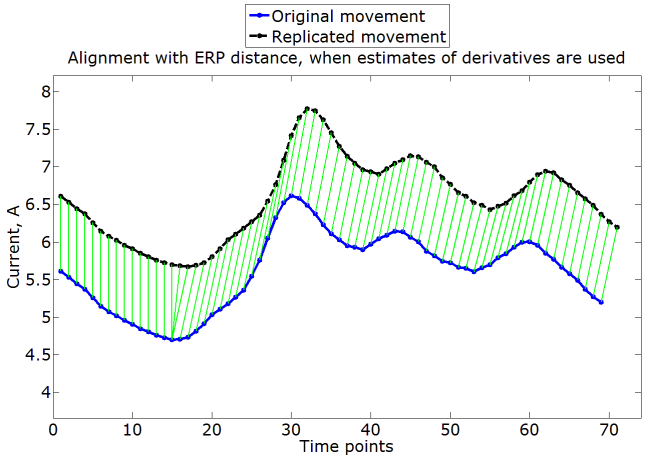


Figure 2.28. ERP alignment on data smoothed with exponential smoothing

In this case, all of the points were aligned perfectly after smoothing was applied. The fluctuations in the time series that existed around time point 50, in Figure 2.27, were smoothed and thus the ERP produced a better alignment of these points. The effect of different smoothing parameter values will be analysed in terms of OCSVM classification accuracy in section 2.5.4.

2.4.3.4 Down sampling

One of the simplest ways to speed up the calculations of similarity measures is the down sampling of the original time series. There are some advanced methods to perform the down sampling like Principal Component Analysis (Pearson, 1901), Discrete Wavelet Transform (Haar, 1910) etc. In this study a simple technique is chosen – linear down sampling, which was applied to speed up the recognition of hand written characters in (Vuori et al., 2001). If a sequence of real valued numbers $A = (a_1, a_2, \dots, a_N)$ is considered, then linear down sampling can be performed in the following way: keep the first and then every k-th data point from the original sequence A and abandon the intermediate data points, where $k = 1,2, \dots,6$ is the down-sampling factor. Consider a simple example, when down sampling factor $k = 3$ and the original sequence $A = (12,5,6,7,10,11,15)$. The resulting down sampled sequence is $\hat{A} = (12,7,15)$. $k = 1$ means that no down sampling is performed. Due to the down sampling of the original sequence, calculations of the ERP distance on the down sampled sequence should be performed more quickly, however some loss of the accuracy in the alignment is expected. The effect of different down sampling factors will be analysed in terms of OCSVM classification accuracy in section 2.5.4.

2.4.3.5 Estimates of the derivatives

As Keogh and Pazzani showed in (Keogh and Pazzani, 2001), a better alignment for two time series is given, if the DTW algorithm is applied on the estimates of the derivatives, instead of absolute values. For the time series $A = (a_1, a_2, \dots, a_N)$, the derivative for a data point a_i was defined as follows:

$$D(a_i) = \frac{(a_i - a_{i-1}) + \left(\frac{(a_{i+1} - a_{i-1})}{2}\right)}{2}, 1 < i < N, \qquad \textbf{2.40}$$

where a_i is the ith element of time series, N is the total length of time series and $D(a_i)$ is the estimate of the first derivative in point a_i.

The estimate is not defined for the first and the last elements of the time series, thus the first element is the same as the second one and the last is the same as the penultimate one:

$$D(a_1) = D(a_2), \qquad D(a_N) = D(a_{N-1}). \qquad \textbf{2.41}$$

The alignment of two time series when evaluating similarity measures and using estimates of the derivatives, instead of absolute values, is expected to be better. A simple example is given to demonstrate the effectiveness of the approach. Using a chosen time series of measurements of current a shift across the x-axis and two bumps along the y-axis are added to obtain a replicated time series. The original and replicated time series can be seen in Figure 2.29. The shift was added at time point 15 (as in the example in Figure 2.17) and a peak was increased from time point 26 to 33 and the depth of a valley was increased from time point 46 to 60.

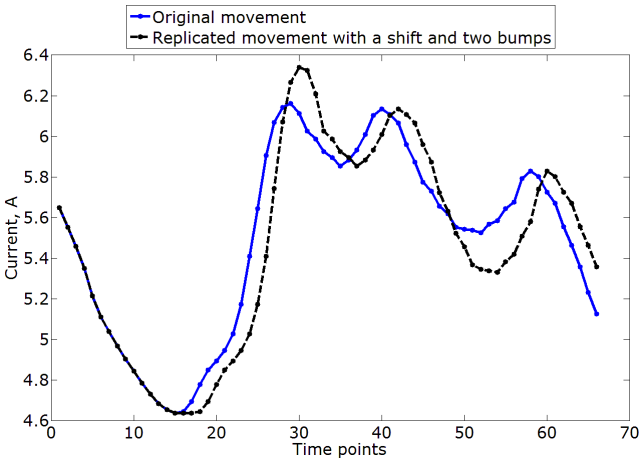


Figure 2.29. Artificial problem for time series matching with ERP

The perfect alignment of the two time series is shown in Figure 2.30. To visualise the alignment, the replicated movement is shifted up in the plot. As previously, the alignment of two time series is given by lines between two points:

- Red line indicates that the alignment of two points is not perfect.
- Green line indicates that the alignment of two points is perfect.

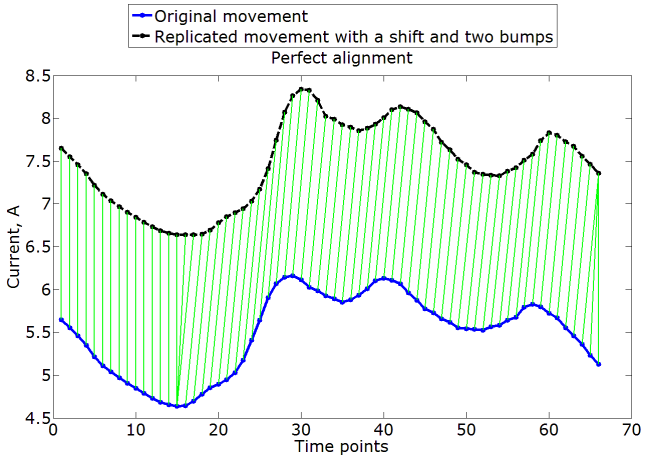


Figure 2.30. Perfect alignment for the artificial problem

The alignment of the original and replicated time series given by DTW similarity measure is shown in Figure 2.31. It can be observed that DTW applied on absolute values was unable to deal with the deepened valley and it produced a lot of one to many matching for time point 53. Less time warping was made when DTW was applied on estimates of the derivatives, as seen on the right hand side of Figure 2.31, however, there still was some unnecessary warping and matching between points that differed from the perfect alignment.

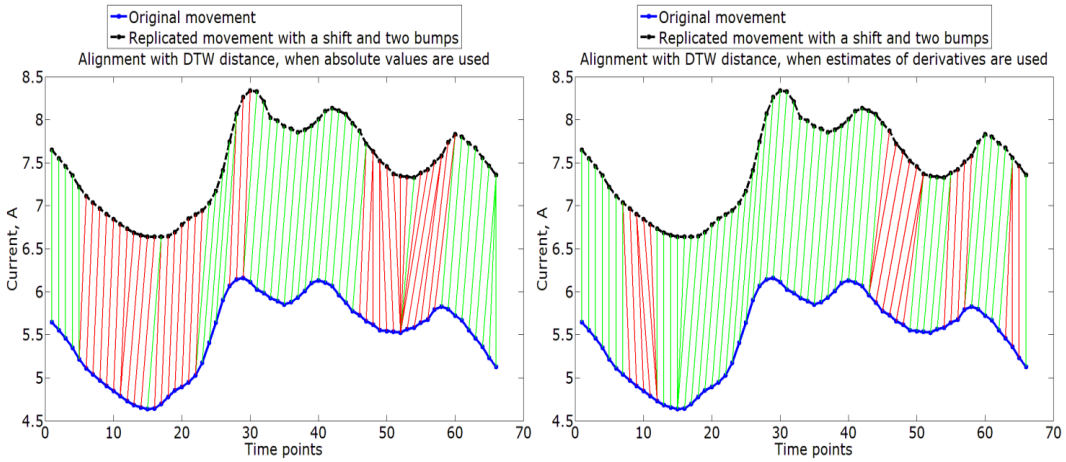


Figure 2.31. Alignment given by DTW based on the absolute values (left figure) and estimates of the derivatives (right figure)

The alignment obtained by ERP metric is plotted in Figure 2.32. When ERP was applied on absolute values, the matching given for the two time series was very poor, especially for the points lying in the deepened valley. In several occasions points of one time series were matched with a long sequence of points from other time series, and it failed to match the valleys properly. When ERP was applied on the estimates of the derivatives, as shown on the right hand side of Figure 2.32, a much better alignment was given. Little unnecessary warping was produced and the matching of valleys increased significantly.

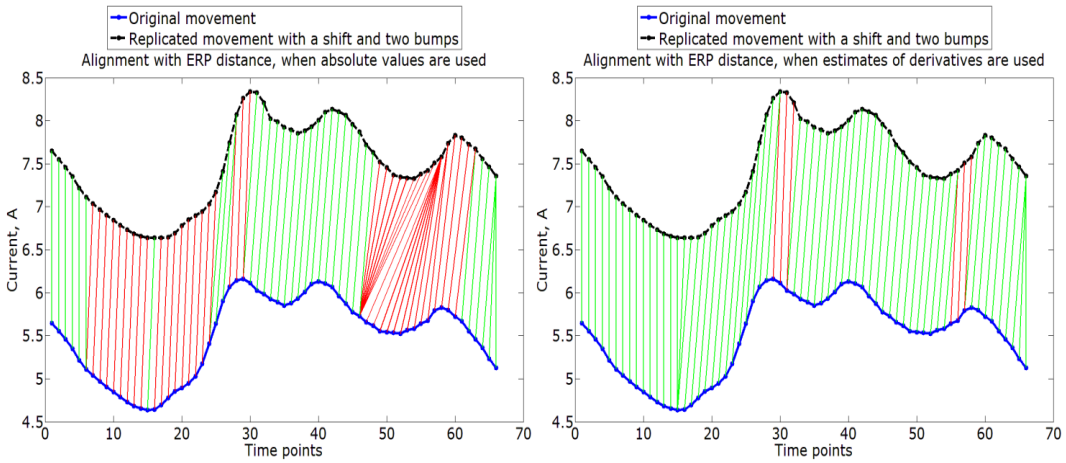


Figure 2.32. Alignment given by ERP based on the absolute values (left figure) and estimates of the derivatives (right figure)

Less unnecessary warping was performed by ERP method (on the right hand side of Figure 2.32) than with DWT (on the right hand side of Figure 2.31). There were only a few points which were aligned differently from what was supposed to be a perfect alignment (red lines indicated deviation from the perfect alignment).

Overall, the ERP method managed to deal with this artificial problem better than the DTW method and the usage of estimates of derivatives helped both methods to provide better alignments, than the usage of absolute values. Thus, the introduction of the estimates of derivatives helps making better alignments when there exists some scaling not only in the x-axis, but in the y-axis as well. The usage of estimates of the derivatives is expected to increase the classification accuracy of the OCSVM, since it was shown that the alignments between time series were improved. The effect of using absolute values or estimates of the derivatives for the matching of time series will be analysed in terms of OCSVM classification accuracy in section 2.5.4.

2.4.4 Calculation of similarity measures

Similarity measures were pre-calculated with all the different pre-processing options, before they were used in OCSVM kernel. The reasoning behind this was that to test the performance of OCSVM, different OCSVM parameter values had to be used and for each set of parameter values the similarity measures would be needed to be recalculated. However, the similarity measure values do not depend on the OCSVM parameter values, thus, calculating similarity values in advance offers a great time saving advantage for extensive training and testing performed in the evaluation of the OCSVM performance.

Actually, pre-calculated values can be stored in a dataset and only values of relevant time series can be picked out for each training and testing dataset instead of recalculating the similarity measures each time a new OCSVM model is built. This is especially useful when elastic similarity measures are used, since they take more computational operations to complete than the standard Euclidean distance. Thus for each combination of different pre-processing techniques the Euclidean, ERP, ERP combined with Euclidean, DTW and DTW combined with Euclidean distance similarity measures were calculated first before inputting them to the model.

2.4.5 Training and testing of OCSVM

OCSVM, as with most data-driven approaches, needs a well-defined training dataset to perform at its best. The dataset must therefore consist of a representative variety of fault-free movements. They should represent the whole variety of fault-free movements as well as possible, because the classification rate of OCSVM not only depends on chosen parameters of OCSVM but also on the training dataset given. The dataset structure was $\{(\boldsymbol{x}, \boldsymbol{L^t})\}$, where $\boldsymbol{x}$ is an input vector and $\boldsymbol{L^t}$ is a target vector, as defined in section 2.4.2.

The training of OCSVM is performed by optimising Lagrange multipliers in the constrained optimisation problem (Haykin, 2008). The sequential minimal optimisation algorithm implemented in Matlab LIBSVM package (Chang and Lin, 2011) was used for this purpose. The training was completed in two phases. In the first phase distances with elastic metrics and the Euclidean distance were pre-calculated for all the movements. In the second phase the OCSVM was trained to select the support vectors and form the separating boundary between normal and abnormal classes. This kind of approach allowed the pre-calculated similarity measures to be used to optimise the OCSVM parameters with a grid search, which will be described in section 2.4.5.3. This two-phased training approach greatly speeded up the training process, particularly when elastic metrics were used as similarity measures. Two different approaches of time series similarity measure usage was performed as explained in sections 2.3.7 and 2.3.8.

During the testing phase the accuracy of the trained OCSVM model was assessed in the following way: the final output of the OSCVM model was either -1 if the movement was classified as faulty or 1 if the movement was classified as good. This classification result is then compared to the label given in the $\boldsymbol{L^t}$variable. The fraction of correctly classified movements is calculated to determine the classification accuracy.

The training and testing of the OCSVM was performed using two different approaches:

1. With a modified K-fold cross-validation procedure (Mosteller and Tukey, 1968).
2. By taking x% of first good movements for the training and (100-x)% for the testing.

These two approaches are described with more details in the next sections.

2.4.5.1 Modified K-fold cross-validation

The K-fold cross-validation technique for the estimation of the classifier accuracy was introduced by Mosteller and Tukey in (Mosteller and Tukey, 1968). The idea of this approach is to randomly split the whole dataset into K mutually exclusive subsets, if possible of equal size. Let the whole dataset be denoted as D. Then it is divided into K-folds:

$$D = (D_1 \cup D_2 \cup \ldots \cup D_K), \qquad \textbf{2.42}$$

where K is the number of folds, D is the whole dataset, $\cup$ is the union, $D_1, D_2, \ldots, D_K$ are the mutually exclusive subsets. Then the classification algorithm is trained on $D \backslash D_i$ subset, and the excluded subset D_i is left for testing the classification algorithm.

This procedure is repeated until all K subsets, $i = 1,2,\dots,K$ are used in the testing phase once and the accuracy of the classification algorithm can be determined.

Due to the high imbalance between the data in each class, using only the total classification accuracy of the OCSVM method might give misleading results. For example, if there were 400 good movements and 20 faulty movements, then, in the case of 5-fold cross-validation, 320 good movements would be used to train the OCSVM and 80 good movements and 20 faulty movements to test the classification accuracy. If all the 80 good movements were classified with 100% accuracy and all 20 faulty movements were misclassified (0% accuracy), looking only at the total accuracy, the classification rate would still be 80% (80 of 100 movements were classified correctly). Thus the correct classification of good movements would overwhelm the poor classification of faulty movements and produce misleading classification accuracy.

To deal with this issue, the classification accuracy of good and faulty movements in this study was evaluated separately. Only the data of movements identified as fault-free was divided into K folds (all data is divided into K folds in the original algorithm). K-1 folds (or parts) of data of fault-free movements were used to train the model and 1 fold, followed by all the movements with faults, was used for testing.

The classification accuracy in each ith fold of fault-free and faulty movements, $P_G(i)$ and $P_F(i)$ respectively, was evaluated in this study by equations **2.43** and **2.44**.

$$P_G(i) = \frac{1}{N_{Gi}} \sum_{j=1}^{N_{Gi}} I\left(LG_j^c\right) \cdot 100\%, \qquad \textbf{2.43}$$

$$P_F(i) = \frac{1}{N_F} \sum_{j=1}^{N_F} I\left(LF_j^c\right) \cdot 100\%, \qquad \textbf{2.44}$$

where $P_G(i)$ is the classification accuracy of fault-free movements in the ith fold, N_{G_i} is the total number of fault-free movements in the ith fold, LG_j^c is the label given by the classification algorithm to a jth fault-free movement, $P_F(i)$ is the classification accuracy of movements with faults, when the ith fold of fault-free movements is used for testing, N_F is the total number of movements with faults, LF_j^c is the label given by the classification algorithm to a jth movement with fault and I is the indicator function defined by equation **2.45**.

$$I(L^c) = \begin{cases} 0 \; if \; L^c \neq L^t \\ 1 \; if \; L^c = L^t \end{cases}, \qquad \textbf{2.45}$$

where L^c is the label given by the classification algorithm and L^t is the true label.

The K-fold cross-validation accuracies of fault-free and faulty movements, P_G and P_F respectively, were obtained using equations **2.46** and **2.47**.

$$P_G = \frac{1}{K} \sum_{i=1}^{K} P_G(i), \qquad \textbf{2.46}$$

$$P_F = \frac{1}{K} \sum_{i=1}^{K} P_F(i), \qquad \textbf{2.47}$$

where $P_G(i)$ and $P_F(i)$ are as defined in equations **2.43** and **2.44**.

Since the division into K-folds is done randomly, the accuracy of the classification algorithm given by the cross-validation estimate (equations **2.46** and **2.47**) depends on this random division. The best estimate of the K-fold cross-validation would be obtained by going through all of the possible random divisions into folds. The number of different possibilities to divide a dataset of N elements into K equal folds (if such equal division is possible) can be found from:

$$\binom{N}{\frac{N}{K}} + \binom{N-\frac{N}{K}}{\frac{N}{K}} + \cdots + \binom{\frac{N}{K}}{\frac{N}{K}} = \frac{N!}{\frac{N}{K}!\left(N-\frac{N}{K}\right)!} + \frac{\left(N-\frac{N}{K}\right)!}{\frac{N}{K}!\left(N-2\frac{N}{K}\right)!} + \cdots + 1 \qquad \textbf{2.48}$$

where N is the total number of data available, K is the number of folds.

However, this is computationally too expensive to perform. For example, the division of 100 data inputs into 5 folds, with 20 data inputs each, can be done in approximately $5.4 * 10^{20}$ different ways. Repeating the division into folds multiple times should provide a good estimate of the accuracy as suggested in (Kohavi, 1995). However, some researchers ignore the fact of randomness and provide the estimate of cross-validation with one iteration of random splitting as can be observed in (Asada and Roberts, 2013).

5-fold cross-validation was chosen in this study as it is a commonly used approach to assess the accuracy of the classification method (Asada and Roberts, 2013, Gudmundsson et al., 2008). The data of good movements was randomly divided into 5 equally sized sets. 4 sets at a time were used to train OCSVM and 1 set was used to test OCSVM. That way each of the 5 sets was used once for testing purposes. 80% of good movements were in 4 sets used to train the OCSVM, while 20% of good movements and all of the bad movements were used to test it. In this study, the random division was performed 100 times in order to get a better estimate of the classification accuracy and still to remain within feasible time of the whole training and testing procedure. Such random division of data is expected to deal better with the changing behaviour of good movements and a possible influence of weather conditions, since training dataset was randomly sampled from all of the dataset including all four seasons of the year, thus included a bigger variety of good movements.

2.4.5.2 Proportional training and testing

An alternative technique for the evaluation of the classification algorithm accuracy will be used in this study, which is defined as proportional training. The idea of this approach is very simple: take first x% of the fault-free movements for the training phase and leave the other (100-x)% of the fault-free movements for the testing phase. All of the faulty movements will be left for the testing phase, as in the case of cross-validation technique. Since x% of the good movements might not be an integer number, the following equation is used to determine the amount of good movements used for the training:

$$N_G^T = \lfloor N_G \cdot x \rfloor, \qquad \textbf{2.49}$$

where N_G^T is the number of movements used in training, $\lfloor N_G \cdot x \rfloor$ is the floor function, where the result is the biggest integer not greater than $N_G \cdot x$, and x is the percentage of good movements used for training.

As in the case of K-fold cross-validation, the classification accuracies achieved with proportional training for fault-free and faulty movements were calculated separately. The classification accuracies of fault free and faulty movements, P_G and P_F respectively, were obtained using equations **2.50** and **2.51**.

$$P_G = \frac{1}{N_G}\sum_{i=1}^{N_G} I(LG_i^c) \quad \textbf{2.50}$$

$$P_F = \frac{1}{N_F}\sum_{i=1}^{N_F} I(LF_i^c) \quad \textbf{2.51}$$

where N_G is the total number of fault-free movements used for testing, LG_i^c is the label given to the ith good movement by the classification algorithm, I is the indicator function as given by equation **2.45**, N_F is the total number of faulty movements used for testing, LF_i^c is the label given to the faulty movement by the classification algorithm.

This testing technique was chosen due its simplicity: no random division is needed and thus no repetitive retraining of the model is needed to estimate its performance, as in the case of K-fold cross-validation. However, this approach might give poor classification results, if features of data of the tested class changes with time. For example, if the first 20% of good movements are chosen to train and test the OCSVM model and the remaining 80% of the good movements somehow change their behaviour resulting in different shapes and values of the measurements of current, then OCSVM trained with such an approach might consider those movements as outliers and therefore classify them as faulty. It is the biggest disadvantage of this technique if compared to the K-fold cross-validation where movements are randomly chosen from the whole data set and thus should not suffer from a changing behaviour of good movements.

2.4.5.3 Grid search for the optimal values of the OCSVM parameters

The classification accuracy of the OCSVM model, as shown in section 2.3.5, depends on two parameters γ and ν. These two parameters highly influence the classification accuracy of OCSVM. A pair of OCSVM parameters (ν, γ) that gives the best classification accuracy can be obtained using a grid search technique, which is a default choice for such an optimisation, e.g. (Li et al., 2010).

The idea of this algorithm is very straightforward: define the possible values of parameters to be optimised and perform an exhaustive search. For example, if the possible values of OCSVM parameters are $\nu = (0.05,0.1,0.15)$ and $\gamma = (10^{-3}, 10^{-2}, 10^{-1}, 1)$, fix the value of one parameter, e.g. $\nu = 0.05$ and run the model with this value and all the possible values of γ parameter, giving the pairs of parameters $(\nu = 0.05, \gamma = 10^{-3}; \nu = 0.05, \gamma = 10^{-2}; \nu = 0.05, \gamma = 10^{-1}; \nu = 0.05, \gamma = 1)$.

The algorithm used to find the pair (ν, γ) that led to the best classification accuracy of the OCSVM model is summarised in Figure 2.33. Using the algorithm, the expected classification accuracies of fault-free and faulty movements, P_{GE} and P_{FE} respectively, were set and a search for a model, which would satisfy a system of inequalities **2.52**, was performed.

$$\begin{cases} P_G \geq P_{GE} \\ P_F \geq P_{FE} \end{cases} \quad \textbf{2.52}$$

If no model satisfying the inequalities **2.52** was found, the values of P_{GE} and P_{FE} were decreased by 1% and the search was repeated until at least one model was found. The decrease in accuracy of 1% was chosen as a smaller change is considered to be insignificant. If several models were found, the one with the highest difference between the expected and achieved acucracies (see equation **2.53**) was chosen.

$$\max\left((P_G - P_{GE}) + (P_F - P_{FE})\right) \quad \textbf{2.53}$$

The algorithm favours the values of OCSVM parameters that have similar classification accuracies for both classes. Depending on the requirements of the fault detection process, this algorithm can be modified to place more emphasis on improving the accuracy of one class, for example, a higher accuracy of classification of movements with faults.

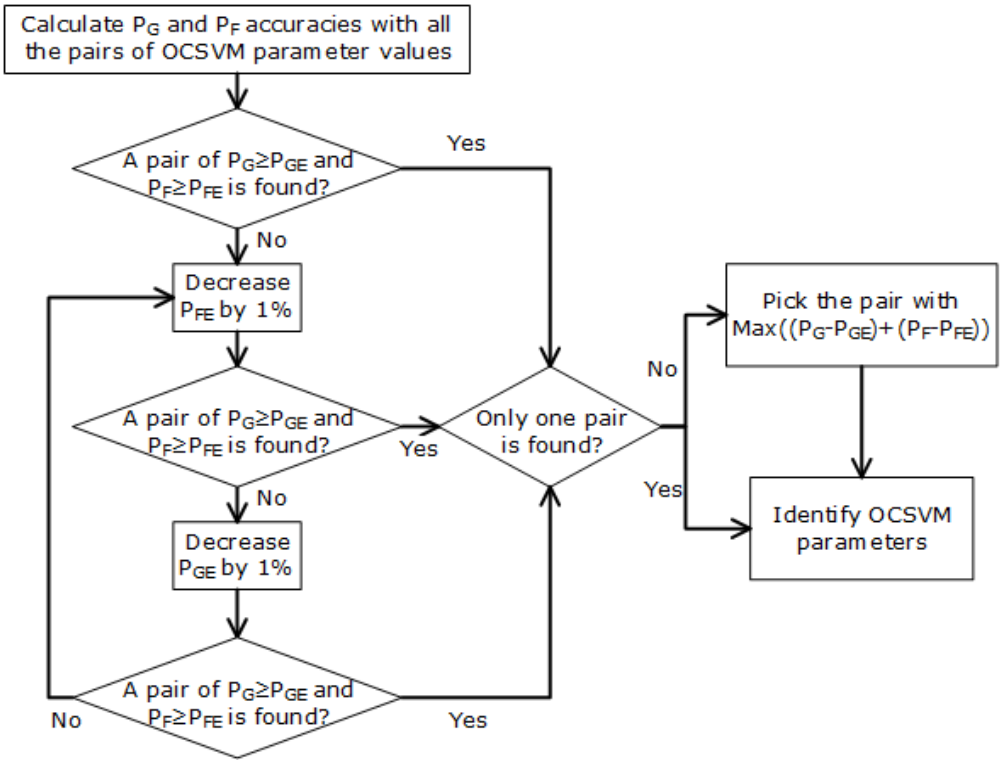


Figure 2.33. Algorithm to find optimal values of OCSVM parameters

The algorithm described above was also used to find the best classification accuracies, for every pre-processing option used separately and jointly (section 2.4.3). The results of the analysis will be presented in section 2.5.

2.4.6 Summary

A proposed fault detection methodology, based on OCSVM and using elastic similarity measures, was presented in this section providing details of each part of the methodology. First an overview of different pre-processing techniques that were tested in this study was given. The expected advantages and disadvantages with certain pre-processing options were identified.

An idea to pre-calculate similarity measures before the training and testing of OCSVM was then presented. This was done to avoid repetitive calculation of similarity measures, this way reducing the time needed for the OCSVM training and testing phases. Two options of different OCSVM training and testing procedures were discussed: 5-fold cross-validation and proportional training. The evaluation of classification accuracy was presented next, detailing a chosen approach how to deal with the big difference of amount of two classes in the testing phase, by simultaneously reducing the expected classification accuracies to find the best one. Finally, a grid search technique was presented to find optimal values of OCSVM parameters γ and ν from a given grid.

In the next section, the application of the proposed methodology is presented and the main results are summarised.

2.5 Application of the proposed methodology to the data of in-field railway point systems

2.5.1 Introduction

Data of point system movements in normal to reverse direction collected from 4 point systems with a clamp lock over a one year period including any alarms and faults registered is used to analyse performance of the proposed approach. The steps of the proposed methodology can be revisited in Figure 2.21. For some of the steps of the methodology, several options were considered. All the different options tested in this study can be summarised in two flowcharts, given in Figure 2.34 and Figure 2.35.

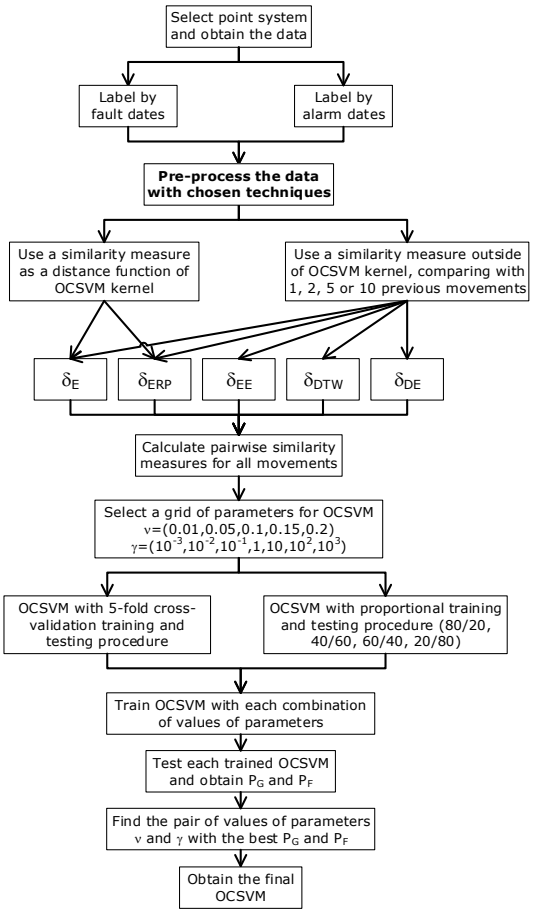


Figure 2.34. Options considered when testing the proposed methodology

In terms of notations used presenting the results, the down-sampling factor will be denoted as k as in section 2.4.3.4. The accuracy of good movements will be denoted as P_G and the accuracy of faulty movements as P_F. The similarity measures used in this analysis will be represented by the symbols that have been described in section 2.3.6. The only difference here will be that if the data was rescaled, it will have a superscript R, e.g. Euclidean distance δ_E^R.

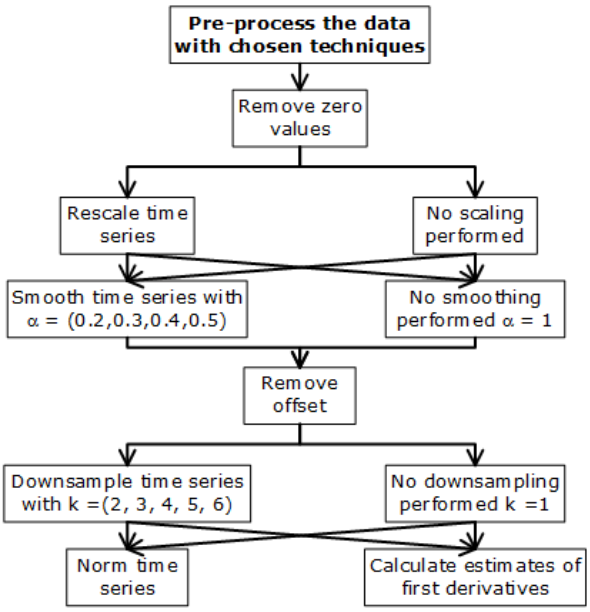


Figure 2.35. Different pre-processing options performed

Given the pre-processing options in Figure 2.35, the number of different combinations of the pre-processing options equals 120 for each similarity measure used, except the Euclidean distance (analysis cannot be performed on non-scaled data as explained in section 2.3.6.2 where the total number of best models found (as described in section 2.4.5.3) for every combination of pre-processing options is 60. The results of each similarity measure are summarised in tables with the layout given in Table 2.3.

Table 2.3. Layout of the presentation of results

Similarity measure	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	Number of models with $P_F \geq 95$ & $P_G \geq 95$	Number of models with $P_F \geq 95$ & $90 \leq P_G < 95$	Number of models with $P_F \geq 95$ & $85 \leq P_G < 90$	Number of models with $P_F \geq 95$ & $P_G < 85$
$90 \leq P_F < 95$	Number of models with $90 \leq P_F < 95$ & $P_G \geq 95$	Number of models with $90 \leq P_F < 95$ & $90 \leq P_G < 95$	Number of models with $90 \leq P_F < 95$ & $85 \leq P_G < 90$	Number of models with $90 \leq P_F < 95$ & $P_G < 85$
$85 \leq P_F < 90$	Number of models with $85 \leq P_F < 90$ & $P_G \geq 95$	Number of models with $85 \leq P_F < 90$ & $90 \leq P_G < 95$	Number of models with $85 \leq P_F < 90$ & $85 \leq P_G < 90$	Number of models with $85 \leq P_F < 90$ & $P_G < 85$
$P_F < 85$	Number of models with $P_F < 85$ & $P_G \geq 95$	Number of models with $P_F < 85$ & $90 \leq P_G < 95$	Number of models with $P_F < 85$ & $85 \leq P_G < 90$	Number of models with $P_F < 85$ & $P_G < 85$

	The desired classification accuracy
	Better than average, but lower than desired classification accuracy
	Average classification accuracy
	Unacceptable classification accuracy

For a better representation of quantitative performance of each similarity measure, the 10 best performing models will be presented with their classification rates P_G and P_F. The results from the application of the proposed methodology are given in the next sections. Only the results for a single point system, denoted PM9A, will be presented, with the remaining results given in appendices.

2.5.2 Data input

As mentioned previously, data from 4 point systems were used in the analysis. Their operations data in normal to reverse and reverse to normal directions were obtained from the databases, presented in section 2.1.6.1. Data from one calendar year of 2012 was selected to include behaviour of railway point system from all seasons and different operating conditions. The amount of data used in this study is summarised in Table 2.4.

Table 2.4. Number of movements used in the analysis

PM ID	N_T	Labelling using alarm dates		Labelling using failure dates	
		N_G	N_F	N_G	N_F
PM9A	589	542	47	531	58
PM9B	575	551	24	520	55
PM13A	519	474	45	478	41
PM13B	521	489	32	480	41

N_T here is the total number of movements for the point system, N_G is the number of movements labelled as good according to the selected labelling strategy and N_F is the number of movements labelled as faulty.

The summary of alarms for PM9A for the time interval considered in this study is given in Table 2.5.

Table 2.5. Alarm records for PM9A

Total movements	**# of movements with alarms**	**# of duration alarms**	**# of peak alarms**	**# of average current alarms**
589	47	44	1	21

Note that one movement of RPS can have several alarms, that is different variables might exceed their thresholds at the same time. That is why the sum of duration, peak and average current alarms exceeds the number of movements with alarms in Table 2.5.

The summary of failures for PM9A is given in Table 2.6.

Table 2.6. Failure records for PM9A

Failure occurrence date	**Failure rectification date**	**Failure cause**	**Maintenance action**
05/02/2012 15:05:51	05/02/2012 19:00:01	Obstruction with snow and ice	Ice removed, points lubricated
25/04/2012 08:41:56	25/04/2012 14:38:01	Obstruction with ballast	Ballast removed
09/05/2012 07:49:14	24/05/2012 15:00:01	Obstruction with squashed cans and bottles, bolts missing from slide chairs	Bolts replaced, obstructions removed, points lubricated
14/09/2012 06:52:00	14/09/2012 11:10:01	Contamination	Contamination cleared, points lubricated
22/11/2012 10:22:11	24/11/2012 04:48:01	Crushed tail cable	Cable replaced

The information about alarms and failures of other point systems are presented in Appendix A.

2.5.3 Results with time series similarity measure as a kernel distance function

2.5.3.1 5-fold cross-validation training and testing technique

As it was explained in section 2.4.5.1, using the 5-fold cross-validation technique the data of good movements was divided into 5 sets. Thus the amount of data (Table 2.4) used for the training and testing phases with 5-fold cross-validation technique can be summarised in Table 2.7.

Table 2.7. Distribution of good and faulty movements in training and testing datasets

PM ID	Labelling using alarm dates				Labelling using failure dates			
	Training		Testing		Training		Testing	
	N_G	N_F	N_G	N_F	N_G	N_F	N_G	N_F
PM9A	433/434	-	108/109	47	424/425	-	106/107	58
PM9B	440/441	-	110/111	24	416	-	104	55
PM13A	379/380	-	94/95	45	382/383	-	95/96	41
PM13B	391/392	-	97/98	32	384	-	96	41

When the total number of movements cannot be equally divided into 5 folds, two numbers are given with a slash. For example, for the PM9A, in some cases 433 good movements were used in the training dataset and the remaining 109 good movements were used in the testing dataset. In other cases, 434 good movements were used in the training dataset and the remaining 108 good movements were used in the testing dataset. Such division was random and was not controlled. Section 2.5.3.1.1 shows results obtained when the movements were labelled according to alarm dates and section 2.5.3.1.2 shows results obtained when the movements were labelled according to failure dates.

2.5.3.1.1 Data labelling according to alarm dates

First of all, the proposed methodology is tested with the data labelling according to alarm dates. As mentioned in section 2.2.4.2, the alarm system is a common technique for detecting faults in the railway point systems. Thus, it is straightforward to compare the results produced by the proposed methodology and the alarm system. Since a simple threshold technique was able to detect abrupt changes in the current, it is natural to expect that a more sophisticated method would perform at least equally well. The results for all the similarity measures are presented next.

The first similarity measure is the Euclidean distance and the results are summarised in Table 2.8.

Table 2.8. Classification rates with Euclidean distance for PM9A (5-fold cross validation, labelling using alarm dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	27	0	0
$90 \leq P_F < 95$	3	7	4	0
$85 \leq P_F < 90$	0	16	1	1
$P_F < 85$	0	0	1	0

More than half of the best models (37 out of 60) managed to achieve better than average but lower than the desired classification accuracy. The majority of the models (58 out of 60) achieved at least average classification accuracy and 2 models had unacceptable classification accuracy.

The results for the ERP similarity measure are summarised in Table 2.9. Both, the results obtained using non-scaled and rescaled data are presented. One could argue that there is no need to use rescaled data for elastic metrics, since it was shown in section 2.3.6.4, that ERP can be used in the case when two time series differ in length. However, the results on rescaled data are also given in order to compare the performance of the Euclidean distance and ERP on the identical data.

Table 2.9. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9A (5-fold cross validation, labelling using alarm dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	6	25	0	0	$P_F \geq 95$	0	46	0	0
$90 \leq P_F < 95$	8	15	2	0	$90 \leq P_F < 95$	2	9	1	0
$85 \leq P_F < 90$	0	4	0	0	$85 \leq P_F < 90$	0	1	1	0
$P_F < 85$	0	0	0	0	$P_F < 85$	0	0	0	0

From Table 2.9, one can see that there were 6 models that achieved the desired classification accuracy, when ERP was used on non-scaled data. 48 out of 60 models managed to achieve better than average but lower than the desired classification accuracy, while there were no models, which would have unacceptable classification accuracy. When ERP was used on rescaled data, slightly worse results were achieved: no models were able to achieve the desired classification accuracy, but none of them had unacceptable classification accuracy as well. Moreover, 57 out of 60 models managed to achieve better than average but lower than the desired classification accuracy. In both cases, using non-scaled and rescaled data, the ERP similarity measure performed better than the Euclidean distance in terms of movement classification accuracy. The results obtained for the remaining 3 point systems are given in Appendix B.

For a detailed quantitative evaluation of performance of similarity measures, classification accuracies of 10 best performing models for each similarity measure are given in Table 2.10.

Table 2.10. Classification rates of 10 best models for each similarity measure for PM9A (5-fold cross validation, labelling using alarm dates)

#	δ_E^R		δ_{ERP}		δ_{ERP}^R	
	P_G	P_F	P_G	P_F	P_G	P_F
1	94.01	**97.83**	**95.24**	96.37	94.31	97.64
2	94.01	**97.80**	**95.69**	96.34	94.28	97.60
3	94.09	96.99	**95.73**	96.21	94.40	**97.49**
4	94.12	96.50	**95.84**	96.19	94.36	**97.17**
5	94.15	**96.32**	**95.84**	96.15	94.17	96.27
6	94.10	96.12	**95.68**	95.11	94.08	**96.20**
7	94.29	95.74	**95.91**	94.97	94.08	**96.19**
8	94.27	95.74	**95.88**	94.73	94.12	**95.93**
9	94.25	95.74	**95.97**	94.47	94.04	**95.77**
10	94.24	**95.74**	**96.01**	94.37	94.13	95.71

It can be observed, from Table 2.10, that the accuracy of classification of faulty movements, $\mathbf{P_F}$, was highest for Euclidean distance. The accuracy of classification of good movements, $\mathbf{P_G}$, was highest for the ERP distance. Nevertheless, the performance of both similarity measures was quite similar and no big differences in the classification accuracies can be observed. All the best performing models managed to achieve classification accuracies higher than 94% for both good and faulty movements.

2.5.3.1.2 Data labelling according to failure dates

The results achieved, when the movements were labelled according to failure dates, are presented in the same way as those obtained using the labelling by alarm dates. The results obtained with the Euclidean distance are summarised in Table 2.11.

Table 2.11. Classification rates with Euclidean distance for PM9A (5-fold cross validation, labelling using failure dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60

Poor classification rates were achieved for all models. The classification accuracy of OCSVM was unacceptable in all 60 cases. The performance of all models changed drastically if compared to the case when the data was labelled according to alarm dates.

The results with ERP distance are given in Table 2.12.

Table 2.12. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9A (5-fold cross validation, labelling using failure dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	60

As in the case of the Euclidean distance, the classification rates of all models fall into the unacceptable classification accuracy level. Such situations can be observed for both non-scaled and rescaled data. Once more, the classification accuracies are very different from the ones obtained with the ERP distance when the data was labelled according to alarm dates. The results for different point systems are given in Appendix B.

In the same manner as in the previous section, the classification rates of 10 best performing models for each similarity measure are given in Table 2.13.

Table 2.13. Classification rates of 10 best models for each similarity measure for PM9A (5-fold cross validation, labelling using failure dates)

#	δ_E^R		δ_{ERP}		δ_{ERP}^R	
	P_G	**P_F**	**P_G**	**P_F**	**P_G**	**P_F**
1	54.55	**59.50**	**58.31**	59.17	51.85	51.20
2	53.25	61.95	**57.15**	**64.86**	56.06	49.34
3	52.13	**61.40**	**57.44**	57.86	50.32	48.73
4	52.22	**61.38**	**60.63**	55.13	48.82	58.36
5	52.07	58.01	**55.63**	**63.16**	48.65	53.50
6	51.87	**63.39**	**55.54**	59.50	48.74	51.32
7	51.61	**61.97**	**58.07**	54.47	48.37	50.74
8	51.42	61.60	**54.68**	**63.30**	50.66	47.34
9	**58.58**	50.22	54.41	**55.54**	56.30	46.71
10	50.13	**64.24**	**57.32**	53.57	50.52	46.38

The results show that not only the classification rates fell into the unacceptable region (P_F<85 and P_G<85), the best performing models were only able to achieve classification rates of around 50%. Such poor classification rates were very different from the ones achieved with the data labelled on alarm dates.

Further investigation of the results with different labelling options will be performed in the next sections to find out the cause of very different classification accuracies. OCSVM performance results obtained with the alternative training and testing technique are presented in the next sections.

2.5.3.2 Proportional training and testing technique

In the next sections the results obtained with the proportional training technique are summarised similarly to those obtained with the 5-fold cross-validation technique. For the sake of brevity, the results of the technique when 80% of data was used for training and 20% for testing are presented in this section. The results obtained with the remaining training and testing proportions are given in Appendix B. Such proportions were chosen to compare this approach with the K-fold technique, since when 5 folds are used, the training data consists of 80% of the dataset. This way the random sampling for the training data and the sequential sampling can be compared. The distribution of movements used for training and testing phases of the proportional training technique is given in Table 2.14 and Table 2.15.

Table 2.14. Distribution of good and faulty movements in training and testing datasets when data was labelled using alarm dates

PM ID	Training 20%	Testing 80%		Training 40%	Testing 60%		Training 60%	Testing 40%		Training 80%	Testing 20%	
	N_G	N_G	N_F	N_G	N_G	N_F	N_G	N_G	N_F	N_G	N_G	N_F
PM9A	108	434	47	216	326	47	325	217	47	433	109	47
PM9B	110	441	24	220	331	24	330	221	24	440	111	24
PM13A	94	380	45	189	285	45	284	190	45	379	95	45
PM13B	97	392	32	195	294	32	293	196	32	391	98	32

Table 2.15. Distribution of good and faulty movements in training and testing datasets when data was labelled using failure dates

PM ID	Training 20%	Testing 80%		Training 40%	Testing 60%		Training 60%	Testing 40%		Training 80%	Testing 20%	
	N_G	N_G	N_F	N_G	N_G	N_F	N_G	N_G	N_F	N_G	N_G	N_F
PM9A	108	434	47	216	326	47	325	217	47	433	109	47
PM9B	110	441	24	220	331	24	330	221	24	440	111	24
PM13A	94	380	45	189	285	45	284	190	45	379	95	45
PM13B	97	392	32	195	294	32	293	196	32	391	98	32

2.5.3.2.1 Data labelling according to alarm dates

First, results with the Euclidean distance are presented, in Table 2.16. Results obtained with Euclidean distance were worse when the selection of training data was done sequentially, rather than randomly sampling it from the whole year data as in the 5-fold cross-validation technique. This time 15 out of 60 models achieved better than average classification accuracy (37 out of 60 models achieved better than average classification accuracy with the 5-fold cross-validation technique). Furthermore, there were 5 models, which achieved unacceptable classification accuracy, while there were 2 such models with 5-fold cross-validation technique.

Table 2.16. Classification rates with Euclidean distance for PM9A (80% training and 20% testing, labelling using alarm dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	7	2	0
$90 \leq P_F < 95$	1	7	3	0
$85 \leq P_F < 90$	24	9	2	0
$P_F < 85$	5	0	0	0

A similar situation has occurred for the ERP distance whose results are summarised in Table 2.17. The number of models which achieved the desirable accuracy dropped from 6 (see Table 2.9) to 3 for the ERP distance on non-scaled data, as can be seen in Table 2.17. However, the number of models which achieved the desirable accuracy for the ERP distance on rescaled data has increased from 0 (see Table 2.9) to 3. The number of models with better than average classification accuracy has increased from 48 to 53 models when ERP was applied on non-scaled data, if compared to the 5-fold cross-validation case. However, the number of such models has decreased heavily for ERP applied on rescaled data (from 57 to 24). Moreover, there were 30 models with unacceptable classification accuracy, when ERP was applied on rescaled data. There were no such models in the case of 5-fold cross-validation. The results obtained for other point systems are given in Appendix B.

Table 2.17. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9A (80% training and 20% testing, labelling using alarm dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	3	17	0	0	$P_F \geq 95$	3	16	0	2
$90 \leq P_F < 95$	27	9	1	0	$90 \leq P_F < 95$	3	5	1	0
$85 \leq P_F < 90$	1	2	0	0	$85 \leq P_F < 90$	1	1	0	2
$P_F < 85$	0	0	0	0	$P_F < 85$	0	0	18	8

The accuracies of the 10 best models for each similarity measure are given in Table 2.18. The best performing models with ERP distance slightly outperformed the best

performing models with Euclidean distance, as can be seen from Table 2.18. The best performing model for ERP distance on non-scaled data managed to achieve P_G = 96.33% and P_F = 95.74%. A similar situation was seen with ERP combined distance on rescaled data, when the best performing model achieved P_G = 95.41% and P_F = 95.74%.

Table 2.18. Classification rates of 10 best models for each similarity measure for PM9A (80% training and 20% testing, labelling using alarm dates)

#	δ_E^R		δ_{ERP}		δ_{ERP}^R	
	P_G	P_F	P_G	P_F	P_G	P_F
1	94.50	93.62	**96.33**	**95.74**	95.41	**95.74**
2	93.58	**97.87**	**95.41**	95.74	**95.41**	95.74
3	93.58	**95.74**	**95.41**	**95.74**	**95.41**	**95.74**
4	93.58	**97.87**	**94.50**	95.74	**94.50**	**97.87**
5	93.58	95.74	**96.33**	93.62	94.50	**97.87**
6	92.66	95.74	**95.41**	93.62	94.50	95.74
7	94.50	91.49	**95.41**	93.62	94.50	95.74
8	**95.41**	91.49	94.50	93.62	94.50	**95.74**
9	93.58	91.49	**94.50**	93.62	**94.50**	**95.74**
10	93.58	91.49	93.58	**97.87**	**94.50**	95.74

In the next section, the results of models when data was labelled according to failure dates, are presented.

2.5.3.2.2 Data labelling according to failure dates

The first similarity measure considered was the Euclidean distance, and the results are presented in Table 2.19. As in the case of 5-fold cross-validation (Table 2.11), the results obtained with all the models are very poor, since all of them fall into the unacceptable classification accuracy region.

Table 2.19. Classification rates with Euclidean distance for PM9A (80% training and 20% testing, labelling using failure dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	1	59

A similar situation can be observed for the ERP distance, whose results are presented in Table 2.20. The classification rates with the ERP distance are very poor and, once more, all of the models fall into the unacceptable classification accuracy region. The results obtained with other point systems are given in Appendix B.

Table 2.20. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9A (80% training and 20% testing, labelling using failure dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	1
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	2
$85 \leq P_F < 90$	0	0	0	4	$85 \leq P_F < 90$	0	0	0	2
$P_F < 85$	0	0	1	55	$P_F < 85$	0	0	0	55

The 10 best models for each similarity measure are given next, in Table 2.21. Results in Table 2.21 just reconfirm that the classification accuracies are very poor, even for the best performing models of any similarity measure.

Table 2.21. Classification rates of 10 best models for each similarity measure for PM9A (80% training and 20% testing, labelling using failure dates)

#	δ_E^R		δ_{ERP}		δ_{ERP}^R	
	P_G	P_F	P_G	P_F	P_G	P_F
1	51.67	46.55	**71.67**	**75.86**	65.00	67.24
2	46.67	46.55	**76.67**	**68.97**	51.67	48.28
3	48.33	44.83	**70.00**	**67.24**	50.00	48.28
4	46.67	44.83	**66.67**	**70.69**	48.33	55.17
5	56.67	43.10	**66.67**	**68.97**	48.33	53.45
6	55.00	43.10	**75.00**	**65.52**	51.67	46.55
7	51.67	43.10	**68.33**	**65.52**	48.33	46.55
8	48.33	43.10	**78.33**	**63.79**	45.00	48.28
9	45.00	43.10	**71.67**	62.07	41.67	**96.55**
10	45.00	43.10	**65.00**	60.34	40.00	**94.83**

Due to such poor results obtained by the labelling data based on failure dates, no matter which training and testing technique was chosen, a more detailed look at how the data was labelled according to the alarm and fault dates is needed. Since, all the conditions for the testing of the two data labelling approaches were the same there must be big differences in how the data was labelled, since the labelling was done in the automated way, just by considering the dates given, as was explained in section 2.1.6.4. In the next section, the time series of the movements that were labelled as good and the ones that were labelled as faulty will be compared visually for the two chosen labelling techniques.

2.5.3.3 Analysis of discrepancies in the data labelling

In this section, the results of the automated movement labelling procedure (section 2.1.6.4) are analysed. Since the classification results obtained with two data labelling approaches were very different, a graphical analysis was performed to identify, which movements were labelled as fault-free and which ones as faulty by both labelling procedures. The labelling given by two different automated labelling approaches was considered for each point system that was chosen for the fault detection analysis. However, in this section, the results were presented only for one point system, since common discrepancies were found for all of the point systems and the remaining example graphs can be found in Appendix C.

In figures in this section the movements that were labelled as good using the alarm or failure dates were plotted in green and the movements that were labelled as faulty were plotted in red. Movements that were labelled based on alarm dates were presented for the analysis only if any discrepancies in the labelling have been visually observed. Other movements were not included in this analysis. All the movements that were recorded faulty according to fault dates were included in this analysis since the classification rate using labelling according to failure dates was very poor.

2.5.3.3.1 Data labelling based on alarm dates

In the first example it can be observed that there were some changes in the data before the alarms have been raised (all the green curves were recorded in the database before the red curves), as seen in Figure 2.36.

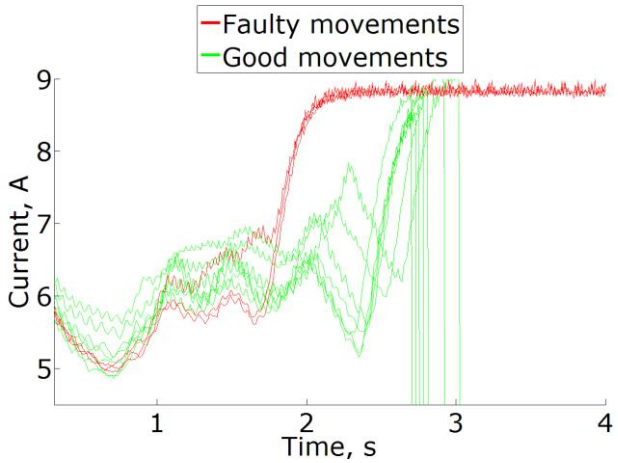


Figure 2.36. First example of imprecise labelling due to alarm dates on PM9A

There were a few good movements (green curves) that had an increase in current, especially after around 2 seconds of the movements. These movements were recorded over a period of three days and then the alarm was raised (red curves), when the POE timed out and did not manage to finish the movement. Note that for the graphical representation, the remainder of these movements were not plotted. Furthermore, this alarm led to an actual fault – obstruction with snow and ice. A possible reason for this fault might be the accumulation of snow and debris throughout several days and the alarm system raised an alarm when only the railway point system was unable to complete the operation. Thus, if the earlier movements that had a big increase in the current values before the alarm was raised were labelled as faulty movements, perhaps a better representation of a good behaviour and a faulty behaviour of POE would be obtained. Thus the movements that showed signs of changing behaviour of measurements of current could be relabelled as faulty (red curves). The suggested relabelling of the data, which would help to include the information of deteriorating condition of the point system, is plotted in Figure 2.37.

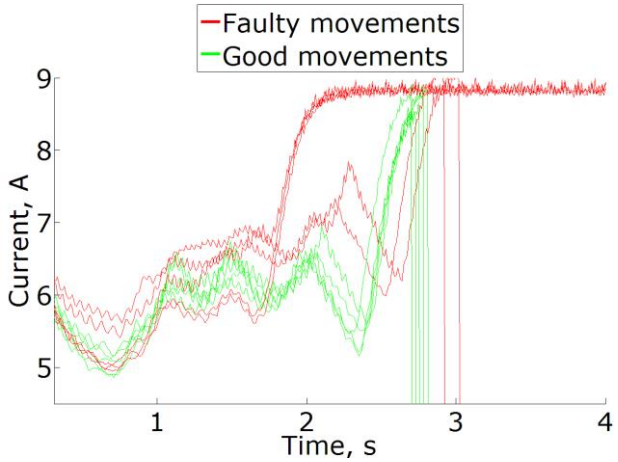


Figure 2.37. Suggested relabelling for the first example

In this first example, the inability of the alarm system to detect changes in the current has been shown, when an earlier detection of that change might have helped to

avoid the failure of the railway point system. In the next example, the inability of the alarm system to detect changes in the current, even when they are abrupt, is shown.

From Figure 2.38, one can notice that there were a couple of movements (in green) that had an increase in current throughout the switch rail movement phase and were close in their profile to the movements that raised the alarm (in red). Moreover, the movements which were labelled as faulty raised an alarm due to the longer motor running duration and the increase of the current values was undetected, since it did not produce an average current alarm. This example shows how the alarm technique is unable to detect abrupt changes in the shape and absolute values of the measurements of current.

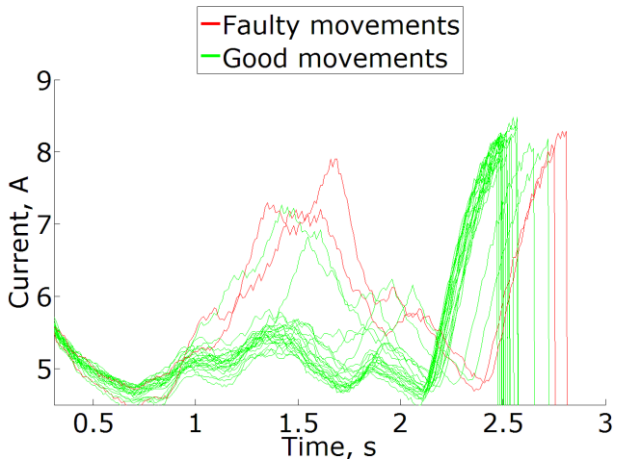


Figure 2.38. Second example of imprecise labelling due to alarm dates on PM9A

Again, an earlier warning can be given if the fault detection algorithm can take account of such changes in current. Thus, the labelling given in Figure 2.38 could be changed in order to include the information of the changing behaviour of measurements of current. The suggested relabelling for this second example is given in Figure 2.39.

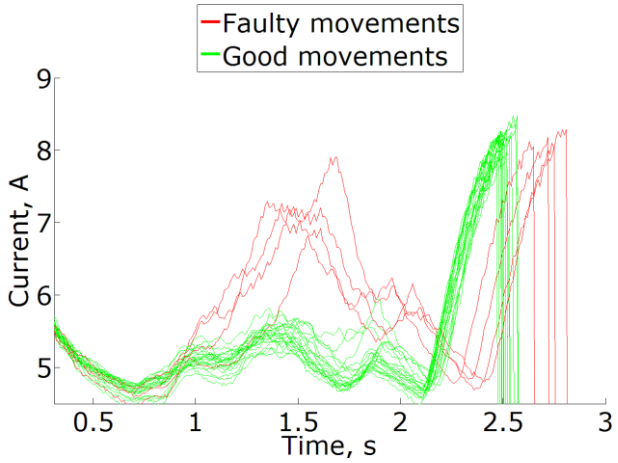


Figure 2.39. Suggested relabelling for the second example

The most common problem found in the process of labelling according to alarm dates is that only the movements that significantly differed from the remaining ones raised an alarm in the system. Moreover, most of the alarms were raised due to the violation of a motor running duration threshold. Even in the case like the ones plotted in Figure 2.38, the alarm was raised due to the longer duration of the movement. However the

differences of absolute values are more than obvious if looking at some of the movement that did not raise an alarm but looked similar to the ones that looked faulty.

Thus if the data is labelled according to alarm dates, only the movements that greatly differ from the other ones would be considered as faulty ones and due to this reason the classification rates are very good. However, no such intelligent techniques were necessary to detect these abrupt changes in the data, since all of them were detected by the alarm threshold technique. Thus, the usage of data labelling according to alarm dates will not allow to train and test the classifiers to detect a prior condition of failure. The detection of conditions leading to faults is one of the key requirements of a good fault diagnostic technique as stated in section 2.2.5 and thus the automatic labelling of data according to alarm dates could not be used for detecting a deteriorating condition of the railway point system.

2.5.3.3.2 Data labelling based on failure dates

As in the previous section, data of one point system labelled PM9A is considered, since similar discrepancies were identified in all of the point systems. The graphs of remaining systems are given in Appendix C. Five faults were recorded for PM9A in the FLD database. Failure occurrence and rectification dates, number of faulty movements, cause of the failure and maintenance actions performed to rectify the failure were summarised previously in Table 2.6.

As in the case of labelling based on alarm dates, graphical analysis was performed to visually compare movements labelled as good and as faulty. 5 movements that were labelled as good prior to the failure date and 5 movements that were labelled as good after the failure was rectified will be plotted together with all movements that were labelled as faulty. Good movements prior to the failure will be plotted in green, faulty movements in red and good movements after the failure was rectified in blue colour.

The first recorded fault of PM9A in the analysis period was identified as obstruction by snow and ice. A rise in current usage is expected when this type of failure occurs, as shown in Figure 2.10. The behaviour of the point system might be unaltered until the moving blades meet the obstruction and two scenarios might occur, as described in (Bolbolamiri et al., 2012) and section 2.1.6.5. The data of the movements is plotted in Figure 2.40.

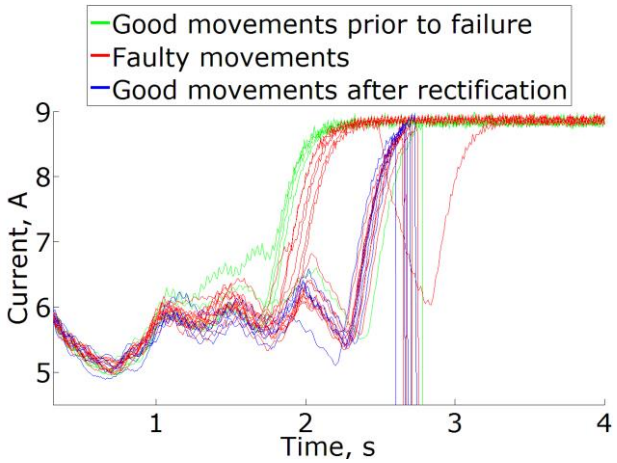


Figure 2.40. Good movements and faulty movements with obstruction by snow and ice

From Figure 2.40 one can see that some movements, which were automatically labelled as good, took more than 6 seconds until the POE timed out and their current measurements were similar to the ones that were labelled as faulty. Moreover, some current measurements of movements, which were labelled as good, are clearly above the ones of red (faulty) movements. However, these movements were labelled as good according to failure dates. This excess of current usage could be the indication of an obstruction by snow, which was the fault cause as identified in the FLD.

In the FLD, the rectification time of the failure was recorded four hours later from the occurrence time. However, the movement from which the increased current usage can be observed was performed three days earlier before the fault date and time given. After this movement, 4 more movements were performed by POE, all of them with the same excess of current usage. During the further 4 movements the point system failed to lock the points and the POE timed out. Only after the points failed to lock, the fault has been recorded. However, the growing excess of current usage in the previous movements should have been a good indicator that a fault might occur in the near future and thus such failures could be predicted. In addition, the movements that were labelled as faulty started behaving as good movements soon after the failure occurred. In total 7 more movements were recorded whose current trends resembled those of good movements until the fault rectification has been recorded in FDL. Note that this point system usually performs one movement within 2-3 hours. This potentially shows that movements are logged in the system, when the engineers are checking whether they managed to rectify the fault. Thus some movements, that are actually fault free, are labelled as faulty ones, if the labelling is done automatically according to failure dates from FLD.

The same issue with the labelling based on failure dates can be observed in Figure 2.41. The records in FLD show that this failure was identified as ballast obstruction. Movements that were labelled as good drew more current from POE compared to the ones that were labelled as bad.

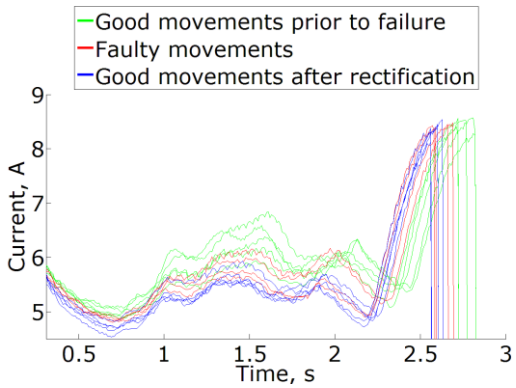


Figure 2.41. Good movements and faulty movements with obstruction by ballast

This type of failure mode, as any obstruction, should reflect a higher current used throughout the rail movement phase and, if the obstruction is not forced out of the way, it might end up failing to complete the movement. The movement plotted in green whose current values were higher than all of the ones of the other movements in Figure 2.41 was recorded on a certain date a couple of days earlier than the failure was logged. It is the same situation as with the previous example in Figure 2.40, when the failure in FLD might have been recorded later than it actually occurred. Moreover, the movements that were labelled as faulty (red curves in Figure 2.41) were almost overlaid by the good movements made after the fault was rectified (blue curves in Figure 2.41). Thus, once more some of the good movements were labelled

as faulty and vice versa when they were automatically labelled relying on the accuracy of the failure dates.

Figure 2.42 shows currents of the point system operating with a broken bolt in the crossing block and obstruction. Once again, the obstruction should result in an excess of current drawn by the POE. However, current from one movement that was labelled as good can be clearly seen to have higher values than currents from any other movements. This could be the movement that actually represented the fault. However it was labelled as a good movement based on the fault data recorded.

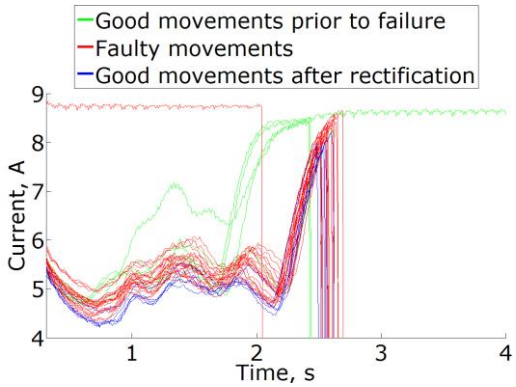


Figure 2.42. Good movements and faulty movements with broken bolt in the crossing block together with obstruction

The movement that was labelled as good but already had a big increase in the current values was recorded 7 hours before the fault was recorded. Three more movements were logged which had an increase in the current values and occurred before the fault was logged. Once more, the findings suggest that the recorded date of the fault might not correspond to the actual date when the fault occurred. Moreover, plenty of movements were labelled as faulty, but they were already similar to the movements labelled as good, after the fault was rectified (plotted in blue), as can be seen from Figure 2.42. As mentioned earlier, this can happen when engineers are testing whether they correctly rectified a failure and whether the fitted alarm system is raising alarms for the rectified movements.

The cause of the failure represented by measurements of current in Figure 2.43 was recorded as contamination. As in the case of dry slide chairs or obstruction, some excess of current usage by POE is expected.

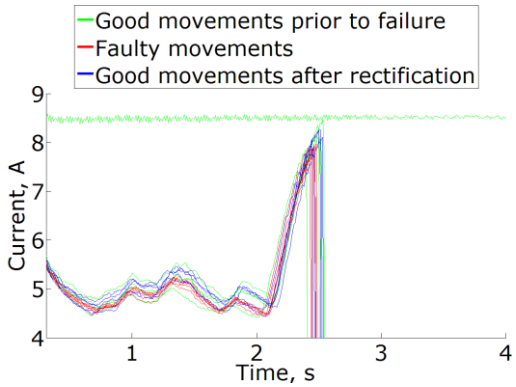


Figure 2.43. Good movements and faulty movements with contamination

One can see in the figure that the movements that occurred before the fault was recorded and, therefore, were labelled as good, actually had higher current values than the movements that were recorded after the failure occurred and were labelled

as faulty. Furthermore, some movements that were labelled as good actually ended up by timing out, when the motor ran for more than 6 seconds. After the first movement that had an increase in current values was recorded, the increase of the current stayed constant for several days in other movements until the point system failed to perform a movement and it timed out. After a few hours of this incident the consequent movement was already recorded as faulty. However, in Figure 2.43 all current trends of the movements labelled as faulty are below the majority of current trends of green and blue movements that were labelled as good. This is not the expected behaviour of faulty system when there is a contamination present.

The last failure was recorded in the database as a crushed cable. The failure data is plotted in Figure 2.44.

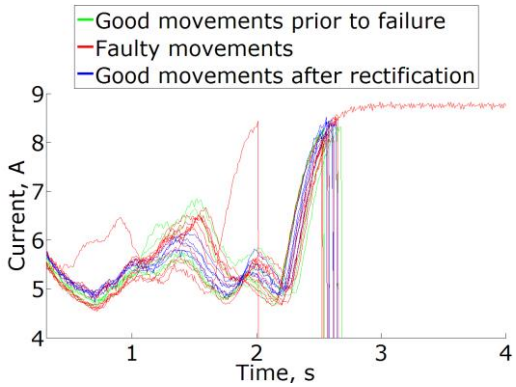


Figure 2.44. Good movements and faulty movements with crushed cable

It can be seen that several good movements (plotted in green) are very similar to the movements, labelled as faulty (plotted in red). However, it is unclear whether the movements were incorrectly labelled as good or as faulty, since it is not clear how the railway point system should behave when such a failure occurs. Nevertheless, there were several very similar movements that were labelled differently.

The most common problem found in the labelling according to failure dates is that the fault actually happens earlier than the specified failure occurrence date. This might be due to the fact that when an alarm is raised by a condition monitoring system which does not disappear after a few movements, only then the engineers are sent out to the field to rectify the situation. The failure date therefore gets recorded later than the date of the first alarms. This way several movements that already represent the failed state are actually labelled as good. Moreover, the failure rectification date is usually recorded later than the date when the system has been fixed. Some movements that occur during the testing of the rectification might be recorded in the movement database and thus they are labelled as faulty despite the fact that the fault was already rectified.

Thus, for these reasons, the automatic labelling of the data according to failure dates becomes misleading. This might explain the poor performance of the classifiers, since some movements are given different labels, but they are actually very similar to each other. With such an inconsistent labelling no classification algorithm could produce good results if no additional information, that would separate those two very similar movements, is given. Therefore, manual relabelling has been carried out and the analysis of the performance of the classification algorithm has been repeated.

2.5.3.4 Suggested data relabelling approach

After taking a closer look at the results of automatic labelling, several deficiencies of both automatic labelling options were found, as explained in the previous sections, signifying a need for a different data labelling approach. However, with the data available, the only other relabelling option possible is to look at each of the individual trends to see whether they are more likely to represent good or faulty movements. Thus a combination of heuristic labelling and labelling using alarm dates and failure dates was performed.

All movements that had alarms were considered to be faulty movements, since the alarms were raised only for current trends that highly deviated from normal ones, as discussed previously. The movements around failure dates were then inspected, identifying which movements still had the alarms, when the alarms disappeared and the POE seemed to be back to normal behaviour. Nevertheless, a visual inspection was additionally performed for each movement of POE PM9A, trying to recognise the movements which represented the deteriorating condition leading to an alarm or failure.

The label of the movement (good or faulty) was determined subjectively, thus the resulting relabelled data indicate which movements looked abnormal (rather than a failed or a non-failed condition). The automation of the relabelling, however, would be impossible, but the cleansing of the data before the model is trained and tested is essential. The results with the relabelled data using the proposed methodology are given in the next section.

2.5.4 Results with relabelled data

After obvious discrepancies were found in both approaches of data labelling, testing of OCSVM where similarity measures were used for feature extraction using automatically labelled data was considered meaningless. A manual relabelling of the data was therefore performed for point system PM9A before carrying out OCSVM performance analysis as discussed in section 2.5.3.4.

This time both options for utilising similarity measures will be analysed: for substituting kernel distance function as described in section 2.3.7 and for feature extraction technique as described in section 2.3.8. The same testing approach has been performed on this relabelled data as in section 2.5.3.

2.5.4.1 Results with time series similarity measure as a kernel distance function

2.5.4.1.1 5-fold cross-validation training and testing technique

Due to relabelling of the data the amount of good and faulty movements has changed and it is summarised in Table 2.22.

Table 2.22. Distribution of good and faulty movements in training and testing datasets

PM ID	Training		Testing	
	N_G	N_F	N_G	N_F
PM9A	385/386	-	96/97	107

The classification accuracy with the relabelled data is evaluated in the same manner as for labelling using alarm or failure dates. The results with the Euclidean distance are presented first in Table 2.23. None of the models were able to achieve either the desired classification accuracy, or better than average classification accuracy. 35 out of 60 models achieved average accuracy. 25 models fell into the unacceptable accuracy region.

Table 2.23. Classification rates using Euclidean distance on rescaled data for PM9A (5-fold cross validation, relabelled data)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	25	4
$85 \leq P_F < 90$	0	0	10	10
$P_F < 85$	0	0	2	9

The results for ERP distance are given in Table 2.24. As with the Euclidean distance, using the ERP distance (Table 2.24), none of the models achieved the desired classification accuracy. The number of models that achieved the average classification accuracy was similar to that achieved in the case with Euclidean distance. 38 out of 60 models when non-scaled data was used and 35 out of 60 models when rescaled data was used achieved the average classification accuracy. However, in both cases (non-scaled and rescaled data) the number of models with unacceptable classification accuracy was quite high (22 and 25 respectively).

Table 2.24. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9A (5-fold cross validation, relabelled data)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	3
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	12	3
$85 \leq P_F < 90$	0	0	38	6	$85 \leq P_F < 90$	0	0	23	7
$P_F < 85$	0	0	10	6	$P_F < 85$	0	0	2	10

The results of the best 10 models for each similarity measure are summarised in Table 2.25. As can be seen from the table, some of the best models are very close to average classification accuracies, however, none of them managed to achieve such accuracies ($90 \leq P_G < 95$ and $90 \leq P_F < 95$).

Table 2.25. Classification rates of 10 best models for each similarity measure for PM9A (5-fold cross validation, relabelled data)

#	δ_E^R		δ_{ERP}		δ_{ERP}^R	
	P_G	P_F	P_G	P_F	P_G	P_F
1	87.68	**93.97**	**89.02**	88.62	88.88	90.70
2	87.25	**94.14**	**89.05**	88.44	88.33	90.86
3	87.48	**93.68**	**88.47**	88.72	88.38	90.00
4	87.32	**93.79**	88.43	88.58	**88.56**	89.64
5	87.06	**93.97**	88.38	88.62	**88.40**	89.18
6	87.18	**93.84**	**88.86**	88.01	88.69	88.31
7	87.12	**93.77**	**88.80**	88.07	88.21	88.57
8	**89.47**	86.46	88.69	88.10	88.43	**88.31**
9	**89.08**	86.33	88.95	87.97	88.02	**88.72**
10	**89.22**	86.01	89.02	87.38	88.52	**88.13**

The performance of Euclidean distance only was quite similar to the ERP distance. This can be explained by looking at the data of the RPS used for the analysis. As mentioned before, uniform scaling was performed on the data in order to be able to use Euclidean distance. Non-scaled data and data after uniform rescaling are plotted in Figure 2.45.

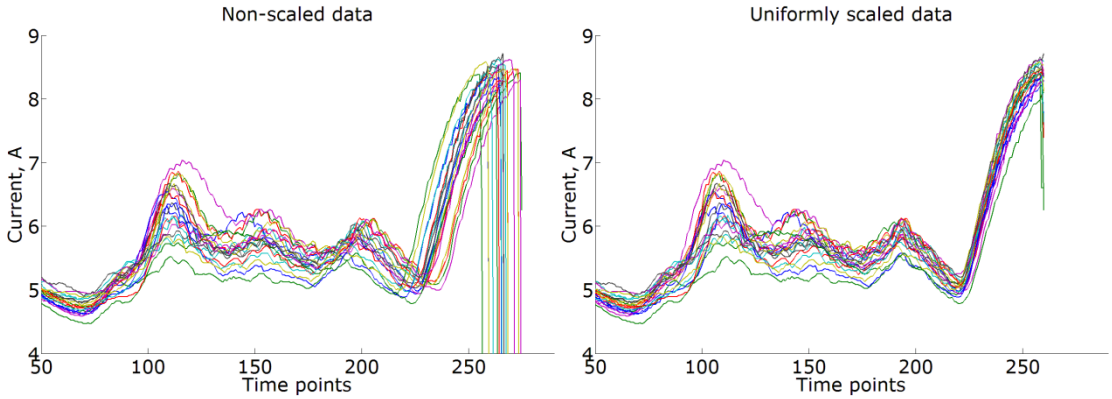


Figure 2.45. Non-scaled (on the left hand side) and uniformly scaled (on the right hand side) data

One can observe that peaks and valleys in the time series even before the rescaling were matched almost perfectly. Furthermore, after the uniform rescaling was applied, these peaks and valleys were even better aligned. Thus in such case the Euclidean distance gives the desired alignment of two time series and performs similarly to elastic metrics. However, if these peaks and valleys would not match, poor results might be achieved using Euclidean distance.

The classification accuracies with relabelled data are worse than the ones obtained using data labelled by alarm dates (Table 2.10). However, it is not feasible to compare these accuracies for several reasons. The first reason is that the training and testing datasets differed in sizes for both cases. 434 good movements were used in the training dataset when the data was labelled on alarm dates, while 385 good movements were used for the relabelled data. Furthermore the testing of the OCSVM model was made with 108 good movements and 47 faulty movements, when data was labelled on alarm dates, while 96 good movements and 107 faulty movements were identified in the relabelled testing dataset.

Moreover, when the data was labelled using alarm dates, only the movements that were very different from normal ones were labelled as faulty as shown in section 2.5.3.3.1. While for the relabelled data, movements that had signs of deteriorating behaviour before an alarm was raised were labelled as faulty. This way the OCSVM models using relabelled data has to deal not only with a smaller dataset in the training phase, but also has a more challenging dataset for testing phase. Thus, the comparison of classification rates achieved with data labelling using alarm dates and with relabelled data is very misleading and thus should not be made. However, the relabelled data is considered to be a better representation of actual condition of the railway point system, thus the results with the relabelled data is of more interest.

The classification rates obtained on relabelled data are quite promising and a closer look on how the pre-processing techniques affected the results is given next.

Influence of exponential smoothing

The influence of exponential smoothing on the performance of the ERP distance was analysed, since the alignment given by the Euclidean distance does not change in

either case. Note that estimates of derivatives of current values rather than actual values are used with exponential smoothing. Only non-downsampled data was tested to avoid mixing the influence of downsampling and smoothing factors. Only the models with the best classification accuracies were selected as described in section 2.4.5.3. The results are summarised in Table 2.26. A '+' sign means that the classification accuracy obtained with the tested smoothing factor was better than that with $\alpha = 1$. On the contrary, '-' sign means that the accuracy was worse.

Table 2.26. Results of best models with different smoothing parameter values

Similarity measure	$\boldsymbol{\alpha = 0.2}$ **vs** $\boldsymbol{\alpha = 1}$		$\boldsymbol{\alpha = 0.3}$ **vs** $\boldsymbol{\alpha = 1}$		$\boldsymbol{\alpha = 0.4}$ **vs** $\boldsymbol{\alpha = 1}$		$\boldsymbol{\alpha = 0.5}$ **vs** $\boldsymbol{\alpha = 1}$	
	$\mathbf{P_G}$	$\mathbf{P_F}$	$\mathbf{P_G}$	$\mathbf{P_F}$	$\mathbf{P_G}$	$\mathbf{P_F}$	$\mathbf{P_G}$	$\mathbf{P_F}$
$\boldsymbol{\delta_{ERP}}$	+7.65	-1.98	+2.71	+0.38	+2.66	-1.72	+2.68	-3.94
$\boldsymbol{\delta^R_{ERP}}$	+5.65	+9.33	+5.19	+8.29	+5.21	+6.27	+0.71	+7.06

In most of the cases, the classification accuracies were better, when exponential smoothing was applied, as can be seen from Table 2.26. The most drops in the accuracies were for the ERP distance obtained using non-scaled data. However, in all of the cases the decrease in accuracy of faulty movement classification was accompanied by a rise in the accuracy of classification of good movements. The results confirm that the smoothing of time series should be performed before the estimates of the derivatives are calculated, since better alignments of two time series is achieved, as it was shown in section 2.4.3.3.

Influence of downsampling

The influence of downsampling factor on the proposed approach performance was tested for all of the similarity measures. This time no smoothing was performed to get a better view of influence of downsampling factor used separately. The results are presented in Table 2.27.

Table 2.27. Influence of downsampling factor

	Aligned by	$k = 2$ vs $k = 1$		$k = 3$ vs $k = 1$		$k = 4$ vs $k = 1$		$k = 5$ vs $k = 1$		$k = 6$ vs $k = 1$	
		$\mathbf{P_G}$	$\mathbf{P_F}$	$\mathbf{P_G}$	$\mathbf{P_F}$	$\mathbf{P_G}$	$\mathbf{P_F}$	$\mathbf{P_G}$	$\mathbf{P_F}$	$\mathbf{P_G}$	$\mathbf{P_F}$
$\boldsymbol{\delta^R_E}$	AV	+0.09	+0.94	+0.06	-9.56	-0.47	+1.55	+2.26	+0.78	-0.34	+0.67
	EOD	+0.6	+14.54	+5.54	+15.41	+5.24	+17.36	+5.51	+20.73	+5.71	+15.8
$\boldsymbol{\delta_{ERP}}$	AV	+0.09	-0.04	-0.2	-0.89	-0.18	-0.44	-0.38	-2.78	-0.29	-0.48
	EOD	+2.15	-0.16	+7.36	-1.17	+2.21	-1.86	+3.28	-3.75	+2.24	-4.29
$\boldsymbol{\delta^R_{ERP}}$	AV	0	-0.41	+0.05	-0.37	+0.08	-1.33	+0.11	+1.99	-0.08	-3.94
	EOD	+4.42	+6.1	+5.03	+7.74	+5.35	+8.06	+5.99	+10.66	+5.47	+7.49

In some cases, the drops in accuracy were quite significant (-9.56%, -4.29%, -3.94%), but in most cases they were of only 1% order. Moreover, for the ERP distance using rescaled data, when data was aligned by estimates of the derivatives, the classification accuracies tended to be better for a bigger downsampling factor. The speed gain of the downsampling of data is quite significant, as it will be shown in section 2.5.5. Some accuracy loss is expected, however in some cases the accuracy has even increased after downsampling, since the downsampling can also help to reduce the noise in the time series.

Influence of estimates of the derivatives

The performance of OCSVM when estimates of the derivatives were used was tested against the use of normed absolute values. As previously, the results are presented using '+' and '-' signs. Since the performance using estimates of the derivatives is expected to be better when data is smoothed, different smoothing factors are tested as well. The results are presented in Table 2.28.

Table 2.28. Change in classification accuracy when the alignment was based on the estimates of the derivatives instead of absolute values

	δ_{ERP}		δ^R_{ERP}	
α	P_G	P_F	P_G	P_F
0.2	-0.02	-1.06	+0.88	-6.85
0.3	-4.93	+0.5	+0.82	-7.33
0.4	-4.92	-1.96	+1.38	-10.05
0.5	-4.89	-4.42	-2.73	-10.08
1	-7.51	-1.05	-1.9	-8.1

One can observe that classification rates were worse in almost all of the cases when the estimates of the derivatives were used. The interesting fact to notice is that for smoothed data, the difference between using estimates of the derivatives and normed absolute values tend to be smaller. It just confirms the fact that smoothing of the time series is vital for calculating estimates of the derivatives. However, it is unexpected that the use of estimates of the derivatives produced worse results, given how the alignment of two time series improves as it was shown in section 2.4.3.5. Moreover, a better performance of elastic similarity measures on estimates of the derivatives was proved in (Keogh and Pazzani, 2001). In the next section, an analysis of the best performing classifier, when 5-fold cross-validation was performed on relabelled data, is presented.

Analysis of the best classifier

In this section, the best performing classifier, when the 5-fold cross-validation technique was used to test and train the model, is analysed in more detail. The best classification accuracy was achieved when using the ERP distance combined with the Euclidean distance on time series of absolute values of current measurements which were down-sampled with $k = 2$ and smoothing factor $\alpha = 0.2$. The classifier reached P_G=89.02% and P_F=88.62 classification accuracy on average. As mentioned before, if the 5-fold cross-validation is performed only once, the estimate of the accuracy is a random number and it might differ from this classification accuracy, which was obtained after 100 runs. For the visualisation purposes, one execution of the 5-fold cross-validation is presented. The OCSVM parameters were $\gamma = 10$ and $\nu = 0.1$. The decision values of the OCSVM method are plotted in Figure 2.46. The colours representing the decision values of OCSVM were chosen as follows:

- The decision values for the movements that were used to train the OCSVM and were identified as good movements are plotted as blue dots.
- The decision values for the movements that were used to train the OCSVM and were identified as abnormal movements are plotted as red dots.
- The movements that were correctly classified in the testing phase are plotted as green dots.
- The movements that were labelled as faulty, but were classified as good, are plotted as red 'x' marks.

- The movements that were labelled as good, but were classified as faulty, are plotted as black '+' marks.

The red line in Figure 2.46 is the boundary between two classes, that is, if the decision function of OCSVM had a positive value, the movement was classified as good and if the decision value was negative, the movement was classified as faulty. The majority of training data was classified as belonging to the good class.

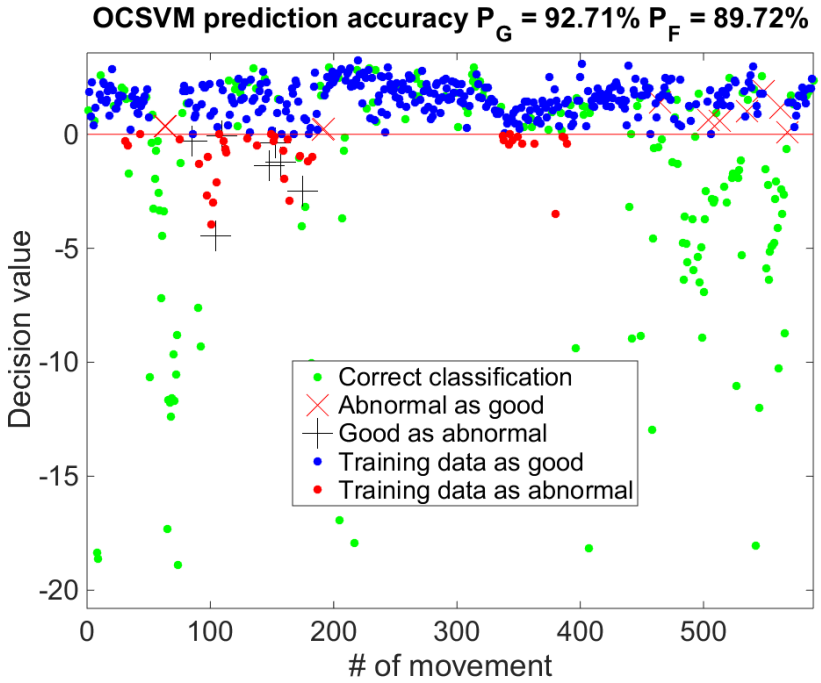


Figure 2.46. Decision values of the OCSVM with 5-fold cross-validation

The summary of classification results is given in Table 2.29, also known as a confusion matrix. The confusion matrix is one of the standard ways of visualising the performance of the classifier. The main idea of the confusion matrix is to show how many movements in different classes were classified correctly and how many movements were misclassified as belonging to another class.

Table 2.29. Confusion matrix

		Classified	
		Good	Abnormal
Actual	Good	89	7
	Abnormal	11	96

From the confusion matrix in Table 2.29 it can be observed that there were 7 movements which were labelled as good but classified as abnormal. Moreover, there were 11 movements which were labelled as abnormal but classified as good. A closer look at some of the misclassified movements is given next. An example of movement which was labelled as good but classified as abnormal is given in Figure 2.47.

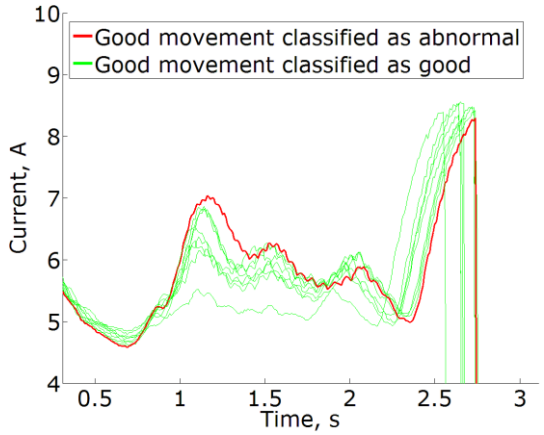


Figure 2.47.An example of a movement labelled as good, but classified as abnormal

The movement, which was misclassified as abnormal (blue line), seems to be above all other movements (green lines), labelled as good and correctly classified, in the interval from 1 to approximately 1.5 seconds, as can be seen in Figure 2.47. Thus, it might be that this movement should have been labelled initially as abnormal, but it is also likely that the OCSVM model produced a false alarm for a good movement. Two examples of movements which were labelled as abnormal but classified as good are given in Figure 2.48 and Figure 2.49.

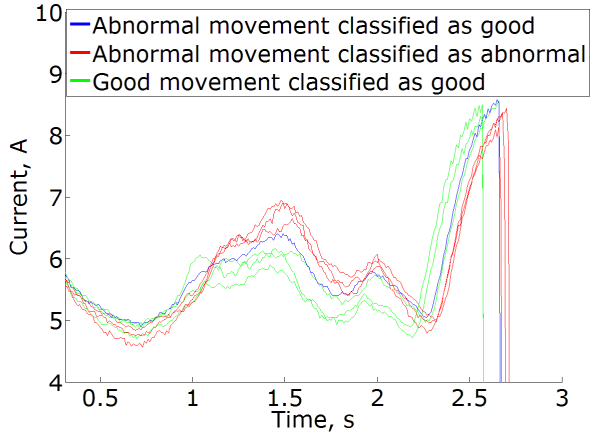


Figure 2.48. First example of misclassified abnormal movement

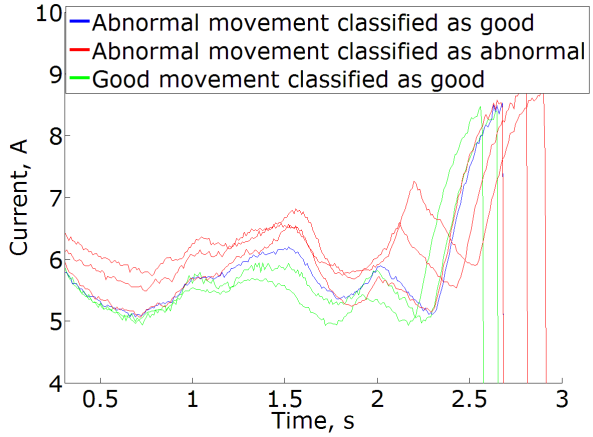


Figure 2.49. Second example of misclassified abnormal movement

In both Figure 2.48 and Figure 2.49, it can be observed that the movement, which was misclassified is in-between the movements classified as good and as abnormal. This might indicate that the movement, which was misclassified, has current values closer to the ones of good movements and only further increase of the current values was classified by the model as abnormal. If such small changes in the profile of current would be classified as abnormal, a lot of false alarms would be raised by the model.

The results demonstrate that the performance of the classifier highly depends on the correct labelling of movements with similar current waveforms. If two very similar movements are labelled differently and if there is no other information which would help to separate them then the classifier would classify them as belonging to the same class.

The performance of this model is next compared with the alarm threshold technique, by considering one example from the testing dataset when the alarms lead to an actual failure of PM9A recorded as crushed cable. In the failure database, the engineers, responsible for the visual analysis of the trends of current measurements, left a note that trends had an increase of current. The movement that raised average current and running duration alarms in the system is plotted in Figure 2.50.

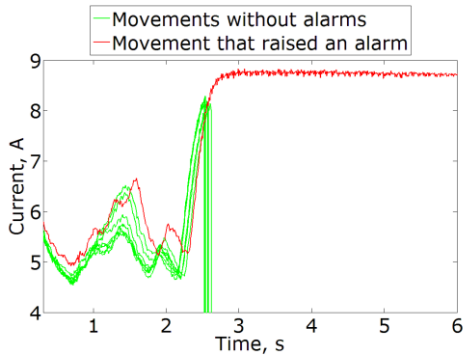


Figure 2.50. Alarm was raised for breaching average current and running duration thresholds

However, if an increase of current is considered, then it can be seen that there were couple of movements, that already had the same increase in current, however only the motor of POE did not time out and thus an alarm was not raised. With the proposed approach, this increase in current is detected, before the motor of POE times out, as shown in Figure 2.51.

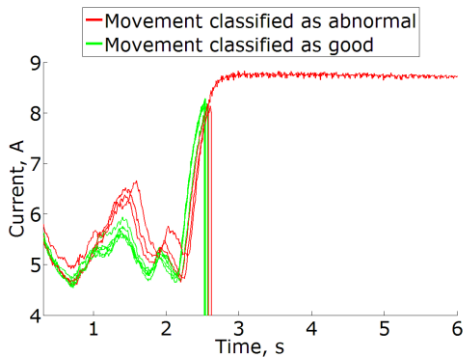


Figure 2.51. Movement classification with the proposed methodology

Thus, changes to the profile of measurements of current can be detected better with the proposed methodology than with the alarm threshold technique. However, it cannot be concluded that the proposed approach is better at fault detection, since changes in the current might not always indicate a fault. Thus more testing using data without the discrepancies in the labelling needs to be performed.

2.5.4.1.2 Proportional training and testing technique

In order to compare the results of the 5-fold cross-validation technique and the proportional training technique, only the case when 80% of the good movements are used for the training is considered next. The numbers of good and faulty movements used for the training and testing of OCSVM using relabelled data are given in Table 2.30.

Table 2.30. Distribution of good and faulty movements in training and testing datasets when data was relabelled

PM ID	Training		Testing	
	N_G	N_F	N_G	N_F
PM9A	385	-	97	107

The results of classification achieved with the Euclidean distance are given first in Table 2.31. 33 out of 60 models achieved a better than average classification accuracy, 23 models achieved the average classification accuracy and only 4 models had the unacceptable classification accuracy. The results are significantly better than the ones obtained with 5-fold cross-validation, where 25 models had the unacceptable classification accuracy.

Table 2.31. Classification rates using Euclidean distance on rescaled data for PM9A (80% training and 20% testing, relabelled data)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	23	10	0	1
$85 \leq P_F < 90$	5	16	2	0
$P_F < 85$	0	1	2	0

The results for the ERP distance are given in Table 2.32. 17 out of 60 models achieved a better than average classification accuracy for ERP using non-scaled data, as can be seen from Table 2.32. For the rescaled data, the number of such models was a bit higher, 22 out of 60. The average classification accuracy was achieved by 12 models using non-scaled data, while 7 models had such accuracy when using rescaled data. However, more than half of the models had an unacceptable classification accuracy (31 for both non-scaled and rescaled data).

Table 2.32. Classification rates using ERP (on non-scaled and rescaled data) for PM9A (80% training and 20% testing, relabelled data)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	5	12	0	1	$90 \leq P_F < 95$	10	12	0	3
$85 \leq P_F < 90$	4	6	2	0	$85 \leq P_F < 90$	2	5	0	5
$P_F < 85$	2	11	10	7	$P_F < 85$	9	3	5	6

Thus, the classification accuracies seem to be better than the ones obtained with 5-fold cross-validation. Such a result was not expected. With 5-fold cross-validation movements for the training dataset were chosen randomly from the whole 1 year period and thus the dataset should have had a greater variability of good movement current trends.

10 of the best performing models for each similarity measure tested in this part of the analysis are presented next in Table 2.33. For all the similarity measures, the 10 best models presented in Table 2.33 had classification accuracies better than 90% (P_G>90% and P_F>90%). The best performing model with the ERP distance using non-scaled data managed to achieve $\mathbf{P_G}$ = 97.94% and $\mathbf{P_F}$ = 94.39%.

Table 2.33. Classification rates of 10 best models for each similarity measure for PM9A (80% training and 20% testing, relabelled data)

#	δ_E^R		δ_{ERP}		δ_{ERP}^R	
	P_G	**P_F**	**P_G**	**P_F**	**P_G**	**P_F**
1	95.88	**94.39**	**97.94**	**94.39**	95.88	**94.39**
2	95.88	**94.39**	94.85	**94.39**	**96.91**	93.46
3	95.88	**94.39**	**98.97**	92.52	96.91	93.46
4	**95.88**	**94.39**	93.81	92.52	**95.88**	93.46
5	**95.88**	**94.39**	92.78	92.52	94.85	93.46
6	**95.88**	**94.39**	93.81	91.59	94.85	93.46
7	95.88	**94.39**	92.78	91.59	**96.91**	92.52
8	95.88	**94.39**	**97.94**	90.65	95.88	92.52
9	95.88	**94.39**	**96.91**	90.65	94.85	92.52
10	**95.88**	**94.39**	95.88	90.65	93.81	92.52

The results obtained when 80% of first good movements were used for training were slightly better than the ones obtained with 5-fold cross-validation. However, this time the Euclidean distance performed similarly to the ERP distance. Given the labelling of the data in this case was adopted to label abnormal looking movements as faulty such results are very promising. Such labelling allows detecting abnormal changes in the current shape and increase in the current values.

Effects of the pre-processing techniques on the performance of OCSVM are considered next.

Influence of exponential smoothing

As in the previous section, the influence of exponential smoothing is determined the ERP distance. The results are summarised in Table 2.34. A '+' sign means that a tested smoothing factor outperformed smoothing factor $\alpha = 1$ (no smoothing), in terms of the classification accuracy. On the contrary, '-' sign means that the tested smoothing factor underperformed.

Table 2.34. Results of best models with different smoothing parameter values

Similarity measure	$\alpha = 0.2$ vs $\alpha = 1$		$\alpha = 0.3$ vs $\alpha = 1$		$\alpha = 0.4$ vs $\alpha = 1$		$\alpha = 0.5$ vs $\alpha = 1$	
	P_G	**P_F**	**P_G**	**P_F**	**P_G**	**P_F**	**P_G**	**P_F**
δ_{ERP}	+11.34	+3.74	+9.28	+1.87	+9.28	+0.93	+12.37	-4.67
δ_{ERP}^R	+6.19	+11.21	+10.31	+7.48	+7.22	+7.48	+6.19	+5.61

Only in one case, the classification accuracy of faulty movements was worse, when smoothing was applied, as can be seen from Table 2.34. In all other cases, the classification accuracies after smoothing was performed increased or remained the same.

Influence of downsampling

The differences of classification accuracies for different downsampling factors are summarised in Table 2.35.

Table 2.35. Influence of downsampling factor

	Aligned by	$k=2$ vs $k=1$		$k=3$ vs $k=1$		$k=4$ vs $k=1$		$k=5$ vs $k=1$		$k=6$ vs $k=1$	
		P_G	P_F	P_G	P_F	P_G	P_F	P_G	P_F	P_G	P_F
δ_E^R	AV	-1.03	0	-1.03	0	-1.03	-3.74	0	+3.74	-3.09	-4.67
	EOD	+22.68	-7.48	+24.74	-5.61	+22.68	-5.61	+20.62	-3.74	+21.65	-5.61
δ_{ERP}	AV	-3.09	-1.87	-12.37	-1.87	-12.37	-1.87	-13.4	-6.54	-14.43	+0.93
	EOD	+12.37	0	+13.4	-0.93	+4.12	-4.67	+12.37	-3.74	+9.28	-8.41
δ_{ERP}^R	AV	+2.06	-7.48	0	0	+2.06	-5.61	0	-13.08	+5.15	-7.48
	EOD	+4.12	+4.67	+4.12	+9.35	+10.31	+4.67	+9.28	+12.15	+6.19	+6.54

As in the 5-fold cross-validation case, the downsampling of the data usually resulted in a decrease of the classification accuracy as can be observed from Table 2.35. However, for ERP distance using rescaled data, the classification accuracies were better, when estimates of the derivatives were used for the alignment of time series. The findings suggest that the downsampling of the data can potentially be performed, if analysis of fault detection has to be performed fast and some loss in accuracy is acceptable.

Influence of estimates of the derivatives

Comparison of the performance results of models when alignments of time series were performed using estimates of the derivatives instead of normed absolute values is given in Table 2.36. As in the 5-fold cross-validation case, the models with the alignment on the estimates of the derivatives performed significantly worse, when data was not smoothed. When data was smoothed with $\alpha = 0.2$, a worse classification of good movements was compensated by a better classification of faulty movements. Moreover, for the ERP distance, the classification accuracies increased when trends of currents were aligned by the estimates of derivatives rather than normed absolute values.

Table 2.36. Change in classification accuracy when the alignment was based on the estimates of the derivatives instead of absolute values

	δ_{ERP}		δ_{ERP}^R	
α	P_G	P_F	P_G	P_F
0.2	+1.03	+14.02	+4.12	+14.95
0.3	-4.12	+12.15	-1.03	+13.08
0.4	+1.03	+9.35	+13.4	+8.41
0.5	+4.12	+3.74	+13.4	-2.8
1	-14.43	+8.41	+7.22	-10.28

Analysis of the best classifier

In this section the best performed classifier, when proportional training technique was used to test and train the model, will be analysed in more detail. The best classification accuracy was reached with the ERP distance combined with the Euclidean distance, when data rescaling was performed. The best accuracy was achieved when two time series were aligned by the estimates of the derivatives, the data was down-sampled with $k = 2$ and and smoothing factor $\alpha = 0.2$. The classifier then reached P_G=97.94% and P_F=94.39 classification accuracies. The OCSVM parameters were $\gamma = 0.01$ and $\nu = 0.2$. The decision values provided by the OCSVM method are plotted in Figure 2.52, using the same colours as those in Figure 2.46, to represent the decision values of different movements.

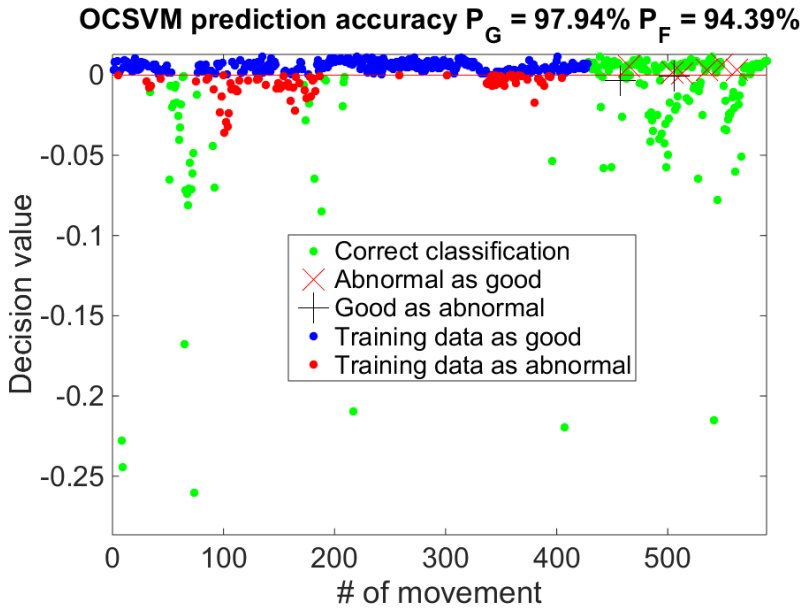


Figure 2.52. Decision values of the OCSVM with proportional training (80% training 20% testing)

What can be seen from Figure 2.52 is that a lot of movements that were used in the training phase were actually considered as outliers by the model (plotted in red dots in Figure 2.52), i.e. most of the good movements used for training were treated as abnormal. This can be confirmed by testing the OCSVM model with the data that was used in the training dataset. From 385 good movements in the training dataset this trained model classifies 83 movements as abnormal, that is the accuracy of classification of good movements in the training dataset is only 78.44%. Nevertheless, the accuracies achieved are better than with the ones achieved with the 5-fold cross-validation approach. However, if more testing with new data was done it is likely that a lot of false alarms would be raised, since even the data which was used for training this model was not classified correctly. Thus, the usage of the proportional training and testing technique might give misleading results and the final model obtained should be analysed in more detail to see whether it is actually valid.

In the next section, the results when the time series similarity measures are used as feature inputs to the OCSVM are presented.

2.5.4.2 Results with time series similarity measure as a feature input to OCSVM

As described in sections 2.3.7 and 2.3.8, time series similarity measures can also be used as feature extraction tools. For this purpose a similarity measure is evaluated for each movement comparing it with several previous movements and then used as inputs to OCSVM with the standard Euclidean distance as the kernel distance function. Due to the deficiencies that were observed in the data labelling, this approach was tested only using the relabelled data. The similarity of each movement with previous 1, 2, 5 and 10 movements was tested in this study. For the sake of brevity, the results of the best performing models when comparing 5 previous movements are presented. The remaining results are given in Appendix D. The 5-fold cross-validation technique was used, since with the proportional training technique the best results were achieved when a lot of good movements from the training dataset were considered as abnormal, as discussed in the previous section. DTW and DTW combined with Euclidean distance were also used for this analysis, since in this case there is no requirement for a similarity measure to be a norm, as it is not used as a kernel distance function. The notation of the new similarity measures followed the same principles as those used earlier.

The results, obtained with all the similarity measures are summarised in one table, since all of the models for all of the similarity measures achieved the unacceptable classification accuracy, as shown in Table 2.37.

Table 2.37. Classification rates using all similarity measures for PM9A (5-fold cross validation, relabelled data)

	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60

As in the previous cases, classification accuracies of the 10 best models for each similarity measure are presented next in Table 2.38 and Table 2.39.

Table 2.38. Classification rates of 10 best models for each similarity measure for PM9A (5-fold cross validation, relabelled data)

#	δ_E^R		δ_{ERP}		δ_{ERP}^R		δ_{EE}		δ_{EE}^R	
	P_G	P_F	P_G	P_F	P_G	P_F	P_G	P_F	P_G	P_F
1	71.27	71.43	**79.59**	56.43	66.68	72.26	79.28	63.31	70.39	**77.81**
2	71.48	71.07	78.33	56.61	66.49	72.38	79.40	62.54	70.98	**76.59**
3	71.07	71.34	78.66	55.79	66.12	72.54	79.31	62.37	70.71	**76.86**
4	71.88	70.94	78.45	55.83	66.08	72.25	79.36	62.23	70.27	**77.26**
5	72.08	70.55	78.55	55.41	66.21	72.07	79.41	61.95	70.81	**76.59**
6	70.70	71.79	78.48	55.44	66.87	71.25	79.45	61.15	70.26	**76.83**
7	70.74	71.69	**79.87**	54.19	66.38	71.00	79.04	61.10	70.68	**76.17**
8	70.24	71.96	79.49	53.69	65.44	72.77	79.51	60.73	69.13	**79.41**
9	70.32	71.70	79.31	53.72	65.68	72.25	**79.58**	60.53	69.38	**78.65**
10	70.48	71.38	**79.53**	53.29	65.03	69.85	**79.53**	59.92	69.43	**78.53**

Table 2.39. Classification rates of 10 best models for each similarity measure for PM9A (5-fold cross validation, relabelled data) (continued)

#	δ_{DTW}		δ^{R}_{DTW}		δ_{DE}		δ^{R}_{DE}	
	P_G	P_F	P_G	P_F	P_G	P_F	P_G	P_F
1	79.25	53.26	79.48	60.99	79.46	63.05	78.68	68.33
2	**79.58**	52.71	79.46	60.50	79.39	62.63	78.82	67.11
3	**79.71**	52.18	79.42	59.66	79.67	62.20	78.84	64.47
4	78.88	51.90	**79.56**	58.73	79.19	62.41	78.57	64.44
5	78.75	51.83	77.36	57.95	**79.47**	60.74	77.42	64.90
6	**79.86**	50.42	76.38	57.97	79.58	60.39	78.88	63.06
7	79.68	50.49	76.24	57.29	79.48	60.15	78.85	63.07
8	**79.76**	50.40	73.62	57.63	79.54	59.66	77.43	63.43
9	79.36	50.75	79.43	56.11	79.43	59.61	77.30	63.32
10	79.41	50.53	77.31	56.98	79.33	59.67	77.13	63.30

None of the models was able to achieve better than 80% classification accuracy. A similar situation was observed when 1, 2 and 10 previous movements were used for the analysis, with the results presented in Appendix D. A number of reasons might have caused such poor classification accuracies.

First of all, when only two consecutive movements are compared and the first abnormal movement after a good movement is detected, but the next movement after the first abnormal movement is also abnormal, then this second abnormal movement would be considered as a good movement, since it is very similar to one previous movement. A similar situation would be observed for 3 consecutive movements. A different situation is observed when more consecutive movements are used for similarity matching. For example, when 10 earlier movements are compared with the one in question and 8 or 9 previous movements are very similar but only one movement is different from normal, this difference might be averaged out with other 8 or 9 previous movements and the abnormal movement left out unnoticed.

Thus, in this case it is necessary to use some additional information about previous movements. Such additional information could be information about whether the new movement has higher absolute values in most of the movement than the movement before or whether the previous movement already differed from normal behaviour etc. However, due to the discrepancies already present in the data labelling, further testing of this approach was left out of this study scope. In the next section, the computational times of calculating similarity measures, training and testing procedures are overviewed.

2.5.5 Results of computational time

Computational times of similarity measures and training and testing procedures are presented in the next subsections. The results are presented for point system PM9A, the one that was used to represent the results of OCSVM performance analysis.

2.5.5.1 Computational time of similarity measures

The results of computational time for the ERP distance (all of the elastic metrics require similar operational time, thus only one similarity measure was chosen) on non-scaled and rescaled data are given in Table 2.40.

Table 2.40. Computational times of similarity measures

k	δ_{ERP}		δ_{ERP}^{R}	
	Average Time, s	**Standard deviation**	**Average Time, s**	**Standard deviation**
1	725.72	4.67	634.70	2.70
2	183.85	1.80	163.92	2.25
3	83.11	0.20	74.77	0.33
4	47.85	0.18	43.44	0.21
5	31.23	0.11	27.93	0.31
6	22.44	0.14	20.60	0.14

With no downsampling, the pairwise similarity measure of 589 movements takes about 726 seconds to complete on non-scaled data and 635 seconds on rescaled data. According to equation **2.35**, calculation of 173166 similarity measures had to be performed, meaning that calculation of two time series similarity measure took around 0.0042 seconds, when data was non downsampled and non-scaled. For the non-scaled downsampled data, when $k = 6$, all the calculations were finished in 22.44 seconds, meaning that calculation of two time series similarity measure took around 0.00013 seconds to complete, that is more than 32 times faster than for non downsampled data. A similar situation was observed for rescaled data. Even downsampling the data with $k = 2$ speeds up the algorithm roughly 4 times, thus the downsampling of the data should be considered if the algorithm is to be used as an online fault detection algorithm and a slight loss in model classification accuracy is acceptable.

2.5.5.2 Computational time of training and testing procedures

The 5-fold cross-validation technique was the most computationally exhaustive part of this study. Since the similarity measures were pre-calculated, as explained previously, the training and testing part of the OCSVM took less time to complete. However, since the simulations were run for 100 repetitive times to get better estimate of the OCSVM parameters, the training and testing procedures, when a grid of 35 different combinations of the OCSVM parameter values (7 values of parameter γ and 5 values of parameter ν) was chosen, took roughly two minutes to complete. That is, 3500 different OCSVM models were trained and tested in 120 seconds, including the data input and output to the model procedures. Thus, one OCSVM model took about 0.034 seconds to be trained and tested.

2.5.6 Summary of the experiments

2.5.6.1 Discrepancies in data labelling

Results that were achieved with two types of automatic labelling procedures were very different. For most of the point systems the classification rates were over 90%, when the labelling of movements was done using alarm dates. In comparison, classification rates of only around 60% were achieved, when the labelling was done using failure dates. This difference in results suggested that in some instances movements that were very similar were labelled differently, when the labelling was done according to failure dates. A detailed analysis of data labelling results with both labelling techniques was performed.

It was found that only significant changes in the value of current raised an alarm and thus, when the data was labelled according to alarm dates, only the movements which were very different from the normal behaviour, i.e. usually the duration of POE operation was significantly longer, were labelled as faulty. Thus it was concluded that data labelling using alarm dates should not be performed, since it does not allow identifying more subtle changes in the profile of current trends, necessary for incipient fault detection.

From a closer look at failure dates, it was found that a lot of movements that represented the already rectified state of the railway point system (therefore, they were similar to the good state) using an automatic labelling approach were still labelled as faulty. Moreover, the failure dates recorded in the database were usually later than the actual failure date, i.e. the failure date was recorded after the engineers found the failure cause, but the movements that were performed prior to the recorded date were already similar to the faulty state. Such discrepancies in the labelling technique using failure dates resulted in poor classification rates and none of the combinations of pre-processing options or different similarity measures used in the analysis helped to achieve better results.

Due to the discrepancies found in both labelling options, the automatic labelling of the data could not be used. In the further study, the movements of one point system were therefore labelled manually, by considering movements that looked abnormal and that looked fault-free. After the manual editing of the discrepancies of the labelling it was shown that the OCSVM model can perform well, as shown in section 2.5.4 and published in (Vileiniskis et al., 2013, Vileiniskis et al., 2015). Ideally, specific RPS engineering knowledge is needed for such data cleansing process, and it would be impractical to relabel a large number of movements.

2.5.6.2 Training and testing procedures

Two different training and testing techniques were used to evaluate the performance of the OCSVM classifiers. The 5-fold cross-validation showed that the results obtained with this technique are more reliable than the ones obtained with the proportional training. It was shown that the best classifier obtained with the proportional training method achieved good classification accuracies by allowing big misclassification rates of the training dataset: a lot of movements representing the fault-free condition of RPS were classified as faulty in the training dataset. Such an approach would lead to a situation when newly recorded good movements would be classified as abnormal if they were similar to the ones used in the training phase. Thus the proportional training and testing technique should not be used to obtain the OCSVM model. However, in order to obtain the OCSVM models with the 5-fold cross-validation method more computational time is required, since the validation needs to be performed repetitively, as discussed in section 2.4.5.1.

If only a single iteration of 5-fold cross-validation is performed to reduce the time, the results might be biased, since the estimates of the classification accuracy depend on the random division of data.

2.5.6.3 Pre-processing techniques

The influence of different pre-processing techniques on classification accuracy has been considered. The use of estimates of the derivatives, in addition to absolute values, for the alignment of data time series did not increase the classification

accuracies in most of the cases, despite the evidence provided in section 2.4.3.5, when it overcomes some weaknesses of the alignment using only absolute values of current measurements.

The classification accuracy usually dropped after downsampling of the data. However, the drop in the accuracy was only around 1% and, as it was shown in section 2.5.5.1, the downsampling can decrease the computational time significantly. If such loss of accuracy is acceptable, the data could be downsampled when the method is applied in practice. Moreover, in some cases the downsampling of data even increased the classification accuracy. Thus the down-sampling might be a good step of the analysis to implement if this approach was to be considered for the online fault detection.

The smoothing parameter had some influence on OCSVM performance, when two time series were aligned by the estimates of the derivatives. Almost in all cases, the accuracy of the model increased if the data was smoothed, when the alignment of time series was based on the estimates of the derivatives.

The effect of rescaling the data was not considered, since the rescaling was only done as an intermediate step to compare the Euclidean distance and elastic similarity measures on identical data.

2.5.6.4 Different ways of using similarity measures

Two approaches of using time series similarity measures were tested in this study. The first approach was to use a similarity measure as a kernel distance function. The second approach was to calculate the similarity measures with several previous movements and then use them as a feature input to OCSVM. Initially, the latter approach was considered to be more useful, since by comparing consecutive movements the model could update the knowledge on the changing behaviour of good movements and in this way seasonal effects could be included in the model. Different numbers of consecutive movements for comparison were considered for the analysis and the comparison of one movement with 5 previous movements proved to be the best from the ones tested. However, using this approach to classify the movement did not provide good classification accuracy. To improve the accuracy of this approach, some sort of weighting of the previous movements could be considered or some additional information used in the model, such as the increase/decrease of values of the measurements of current for every movement, the information about the class of the previous movement etc.

2.6 Conclusions

2.6.1 Introduction

Failures of the railway point systems cause a lot of delays on the UK railway network. Moreover, inappropriate or delayed maintenance of RPS might lead to safety issues and hazardous situations if the poor condition of RPS is not timely detected. A lot of research has been focussed on finding a technique to timely detect the deteriorating condition of RPS. The most common drawback among the proposed methods is the inability to take account of features of in-field RPS data. Weather conditions, usage frequency and forces acting when trains pass the RPS are usually not considered when the data is obtained from the laboratory RPS. The features of in-field RPS data

were overviewed in section 2.1.6 of this thesis. These features were the major factor for choosing an appropriate method for fault detection of RPS used in this study.

In this thesis, several novel aspects were proposed for the fault detection of railway point systems. These aspects will be overviewed in the following sections.

2.6.2 OCSVM as a fault detection technique of RPS

Firstly, the proposed methodology considered fault detection of RPS as a classification problem: classify the movements belonging to a fault-free or a failure class. A special case of the classification algorithm commonly referred to as a novelty or outlier detection algorithm was chosen. The idea of such a method is to detect abnormal data from the majority of good data, when the abnormal data is highly underrepresented. This type of algorithm can also be referred to as one class classification methods.

The use of the OCSVM model was proposed in this study. The OCSVM is an extension of SVM algorithm in the case of unlabelled data. The advantage of these models is that they can be trained only on the data of one class. This is very important when there are big differences in the amount of data available for several classes. In this way, a proportion of data from the good class can be used to train the model and all the data from the abnormal class and the remaining data from the good class can be used to test the trained model. Such a proposed approach was important in this study, since the amount of movements representing a faulty condition of the RPS was very small compared to the amount of movements representing the fault-free condition.

In addition, the OCSVM approach allowed to detect the changing behaviour of profile of current, before large increases in current would be observed and the standard alarm threshold technique would give an alarm. However, the results of the OCSVM approach need to be used carefully, since the changing profile of current might not always indicate a failure of the system, thus more alarms can be raised than with the standard alarm threshold technique. Nevertheless, there might be situations when the knowledge about the deteriorating state of the system, i.e. before a failure occurs, can be useful in planning maintenance to reduce disruptions for the most convenient time of railway operations. Such situations can be detected using the OCSVM model.

2.6.3 Elastic metrics as similarity measures

In this study, each operation of RPS was represented by a time series of measurements of current over time. The similarity of two time series in the classification problems is commonly measured by the Euclidean distance. Due to the operating conditions of RPS, the obtained time series varied in length, as was discussed in section 2.1.6.3.1. Thus the Euclidean distance could not be used on raw data, since this distance cannot be defined for the different length time series. The data had to be transformed in some way in order to be able to use the Euclidean distance. This was done by uniformly rescaling the data to make it of the equal length.

The use of similarity measures which are defined for different length time series was proposed as an alternative to the Euclidean distance for the comparison of two time series of measurements of current. Not only do they overcome the limitation of the Euclidean distance to be used on different length time series, but they also allow a different type of matching, i.e. finding the most similar areas of the time series and

matching them. This elastic matching provides a more realistic alignment of two time series, by matching corresponding peaks or valleys of time series, even if they are not perfectly aligned in time, when measured. The alignment of the time series using elastic metrics is based on absolute values. However, as shown in section 2.4.3.5, such alignments might be improper.

The two time series can be aligned by their shape as proposed by Keogh and Pazzani in Derivative Dynamic Time Warping method (Keogh and Pazzani, 2001): calculate the estimates of the derivatives in each point of the time series and then use the derivatives, instead of absolute values, to match the two time series. But in this case, the information about the differences between the absolute values of the two time series is lost. An approach of combining the estimates of the derivatives and the information of the absolute values, to obtain the similarity of two time series, was proposed in this study. This process can be described in several steps:

- Calculate the estimates of the derivatives for each point in the time series;
- Match each point of the estimates of the derivatives of the time series using the elastic metrics;
- Use the matching provided by the elastic metrics to calculate the Euclidean distance on the original time series;
- Sum the weighted results obtained from matching with elastic metrics and Euclidean distance.

The best performing model in this study was the one using the Edit distance with real penalties. However, the performance of the Euclidean distance was quite similar to the proposed elastic metric. The similar results were obtained because the phases of the POE movements matched almost perfectly after the uniform rescaling of data was done, as shown in Figure 2.45. However, if after the rescaling, the peaks and valleys in the data appeared at different time points, then the alignment provided by the Euclidean distance would probably be worse.

The aim of this proposed methodology was to perform all phases of the fault detection (including training of the models) with minimal need of heuristics. However, the choice of the similarity measure can be made by looking at the data and, in the cases as illustrated in Figure 2.45, Euclidean distance could be used in order to speed up the fault detection process and still obtain similar results as when using the ERP distance.

2.6.4 Testing the proposed methodology using in-field data

The final aspect of novelty of this approach was that the proposed methodology was developed for and tested on data of infield railway point systems. In the research done previously, most of the proposed methodologies were based on the data from laboratory RPS. The main difference between in-field and laboratory data is that the data obtained from the laboratory point system to represent the fault-free movements has very little variance. All movements are performed one after another and thus the obtained measurements are almost identical, which is not the case for infield railway point systems. Every movement of the infield RPS is performed after a certain time interval, for some systems it may be a half of the day, if they are not used frequently, for others it might be 2 hours or 2 minutes. Thus, the behaviour of the railway point system can change with time and there can be a lot of variability even in the non-faulty behaviour of the system.

In addition, the operation of railway point system might also be influenced by severe weather conditions or loads applied to the system which cause variability. When relatively small sample sizes of laboratory data are used to validate the performance of the proposed approaches, the variability of good and faulty behaviour of the system is difficult to replicate. When failures are manually inserted to the system in the laboratory their effects are severe and result in a big deviation from the normal behaviour, which can be easy to detect. The in-field data shows that some failures cause subtle effects to the behaviour of the system, and the proposed method in this study aims to detect such effects.

2.6.5 Impact of labelling deficiencies on the OCSVM performance

The performance of all classification methods highly depends on the correct labelling of the data before it is actually used in the methods. One of the problems, when dealing with in-field RPS data, was that there were deficiencies in both automatic labelling options performed in this study. The summary of the deficiencies was given in section 2.5.3.3. The main problem found was that the labelling, given by the dates of alarms and failures in the databases used in this study, labelled similar looking time series of movements as belonging to different classes. This deficiency was particularly observed using the labelling approach using failure dates. Due to the deficiencies found in the data labelling, the OCSVM performance highly differed for two labelling options.

Very high classification rates were achieved, when data was labelled using alarm dates, with classification accuracies over 90%. On the contrary, the classification rates, when data was labelled using failure dates, were only around 50%. One could argue that the classification rates achieved using labelling on alarm dates, are very good and thus this labelling option should be considered. However, this labelling option does not allow detecting the deteriorating condition of the RPS, since the alarms are raised in the system only when abrupt changes in the measurements of current are detected as discussed in section 2.2.4.2. Thus the correct labelling of data is essential for the proposed approach to be able to perform at its best. Ideally some expert knowledge is required to correctly label the available data.

2.6.6 Practical application of the methodology

The aim of this research was the development of a practical fault detection tool for railway point systems. The majority of the requirements, given in section 2.2.5, for the RPS fault detection tool have been achieved. Each requirement will be briefly overviewed by stating to what extent it has been achieved and what further research is needed.

Able to identify a deteriorating condition prior to a fault occurrence. This requirement was partly fulfilled since it was shown that the changing behaviour of a railway point system can be detected using the proposed methodology. However, as mentioned before, the changing profile or absolute values of current might not always indicate a failure of the system, since it might represent changes occurring due to weather or operating conditions. The testing done with the relabelled data showed that with the proposed methodology a movement was usually classified as faulty earlier than using the alarm threshold technique. However, the testing of the proposed approach suffered from a high number of deficiencies in the labelling of the data. Thus further testing of the approach can be carried out if data relabelling is performed using expert knowledge.

Universal in terms of using data from any type of point system. This approach can be applied to any type of point system, as long as the current trends of POE motor can be obtained. However, the development of the OCSVM model for an individual point system would have to be carried out when a new point system is considered, since optimal parameters of OCSVM have to be found for each RPS considered.

Able to deal with a lack of historical data of faulty conditions. One of the main features of in-field data was a small amount of the failure data available for each RPS. The use of OCSVM solved this issue by using only the data of a fault-free state of RPS to train the model. All the available failure data was then used to test the model. If, however, no historical data of failures was available for the specific RPS, such data series would have to be generated in some intelligent way. It would be feasible to test the model with the data that was generated using expert knowledge about the common types of failures of RPS. In this way, the user of this methodology could test whether the OCSVM model can detect the expected changes in the measurements of current due to certain failures. However, with such an approach there is a risk of inducing changes in the data that are too abrupt or too subtle, thus only an experienced engineer could generate such data. Furthermore, without having any failure data, the training phase of the OCSVM model cannot be automated. Overall, the proposed approach deals with little historical data of faulty conditions, but expert knowledge has to be used, when no data of faulty conditions is available.

Able to detect failure modes that were not present in historical data. Such ability of the proposed methodology comes from the OCSVM model, since the idea of this model is to separate the outliers from the majority of data. In that way, any failure mode that has significant differences from the fault-free state in terms of measurements of current would be detected.

Have a small as possible false alarm ratio. The false alarm ratio (misclassification rate of good movements as abnormal) for the best performing OCSVM model on relabelled data was similar to the misclassification ratio of abnormal movements as good ones. 8 good movements were falsely classified as abnormal, while 100 movements were correctly classified as abnormal, giving the false alarm ratio as 7.41%. 7 faulty movements were falsely classified as good movements, while 88 movements were correctly classified as good, giving the misclassification rate of 7.37%. However, these results were obtained only for one RPS using relabelled data and thus more testing needs to be done to show the efficiency of the proposed approach in other cases.

Have an automated and fast fault detection algorithm, suitable for on-line regime. One of the biggest issues of using the proposed approach in practice is the need to retrain the model if movements of railway point system without a fault change over time, which has been observed in data. Then a good movement potentially would raise a false alarm, since it would not be similar to the other good ones, which were used to train the OCSVM model. The training process requires a lot of computing time to find the best performing OCSVM. Moreover, the need to choose the OCSVM parameters prevents from having a fully automated process to identify the best performing OCSVM model. The choice of parameter ν is easy to define, since it controls the allowed error in the training dataset, and thus for this type of problem a small value ($\nu < 0.1$) should be chosen. However, parameter γ has to be chosen experimentally, for example, using the grid search technique and thus requires extensive computational resources and some heuristics to choose a grid of possible values. This would become a big issue if the model would need to be retrained often, due to the reason described previously.

Moreover, since the changing profile or absolute values of current might not always indicate a failure of the system, the ability of OCSVM to detect this changing behaviour cannot be considered as an automated fault detection algorithm. The process of OCSVM decision about the new movement is automated, however the output of OCSVM would probably have to be reviewed to confirm or reject a failure, since at this stage of the research there was a lack of trustworthiness of the data used in the OCSVM training and testing processes. The proposed approach would be suitable for on-line application, since the computational time required when the training phase of OCSVM is complete is low. To reduce the time needed for the training phase, the downsampling of the data is necessary, since it speeds up the calculations significantly, as shown in section 2.5.5. If expert knowledge on the data available in this study was gained, further development of the methodology could be performed in order to automate the training process.

Some of the requirements introduced for the new fault detection technique have been fulfilled either fully or partially. However, this methodology needs further development in order to be applied in practice, such as the selection of a suitable grid for the optimisation of OCSVM parameters or inclusion of additional information to the model. Moreover, data cleansing is needed, which ideally should be done applying engineering expertise. The possible improvements of the methodology, if engineering support was available, are suggested in the next section.

2.7 Future work

The research performed in this study leaves several future areas to be explored for a better fault detection technique for RPS. One of the options is to extend the use of similarity measures as feature extraction techniques by incorporating additional information to the measurements of current in the OCSVM model, as listed below. Another option could be the division of the measurements of current into phases before the analysis in order to provide some diagnostic capabilities, i.e. to identify which part of the system has failed.

Additional information, such as increasing absolute values of the measurements of current with every movement or detection of abnormal current shapes in earlier movements or weather conditions etc. could be included in the OCSVM model in order to improve the accuracy of fault detection when the similarity measures are used as feature extraction techniques. The use of similarity measures as feature extraction techniques could solve the problem of changing “good” behaviour and seasonal dependencies in the data. Only several consecutive movements need to be compared with such an approach. This way more subtle changes in the data could be detected, since the consecutive movements are expected to be similar. Moreover, the comparison of only a couple of movements reduces the computational time required for the calculation of similarity measures, as discussed in section 2.3.8.

The time series of measurements of current could be divided into operational phases, as shown in Figure 2.2, to provide some diagnostic capabilities. The detection of changes in certain phases would narrow down the number of possible causes of failure. Comparing operational phases separately would also allow more subtle changes in each phase to be detected, since subtle changes in one phase only could be left undetected, when the movement is compared as a whole.

The proposed approach has only been used as a fault detection technique of railway point system. However, fault detection on similar systems could also be performed. The condition of the system should be represented by measurements of some parameters over one operation (e.g. force, current) and the change of shape and absolute values of the measured parameters should indicate a deteriorating condition of the system. Common examples of such systems might be actuators of doors (e.g. automatic train carriage doors, elevator doors) or bridge lifting mechanisms.

3 Study on fault detection and diagnostics for three-phase separators

3.1 Three-phase oil, water and gas separators

The three-phase separator is one of the main components in the oil production plant. This unit is responsible for separating gas, water and solid impurities from oil. The operation of the three-phase separator is based on the laws of gravity, allowing a liquid with a higher density, such as water, to settle on the bottom of the separator, and a liquid with a lower density, such as oil, as well as gas, to flow to the top of the separator. Different types of separators can be used in industry, the most common ones being horizontal, vertical and spherical. A horizontal separator is most commonly used due to the ease of maintenance, good separation quality and low initial set-up costs. In this study, a horizontal three-phase separator with a weir is considered. However, the proposed methodology is generic and with some minor adjustments it could be applied to other types of separators.

In the next sections a commonly used design of the three-phase separator and undesirable events commonly observed during the operation of the three-phase separator are discussed.

3.1.1 System description

A schematic diagram of a typical horizontal three-phase gravity separator with a weir can be seen in Figure 3.1. The whole vessel can be roughly divided into three sections:

1. The gravity settling section (or the liquid separation section), where the separation of water and oil takes place (the section to the left of the weir).
2. The separated oil section, where the separated oil is flowing from the liquid separation section (the section to the right of the weir).
3. The remaining space of the vessel is left for the gas phase (separated gas section).

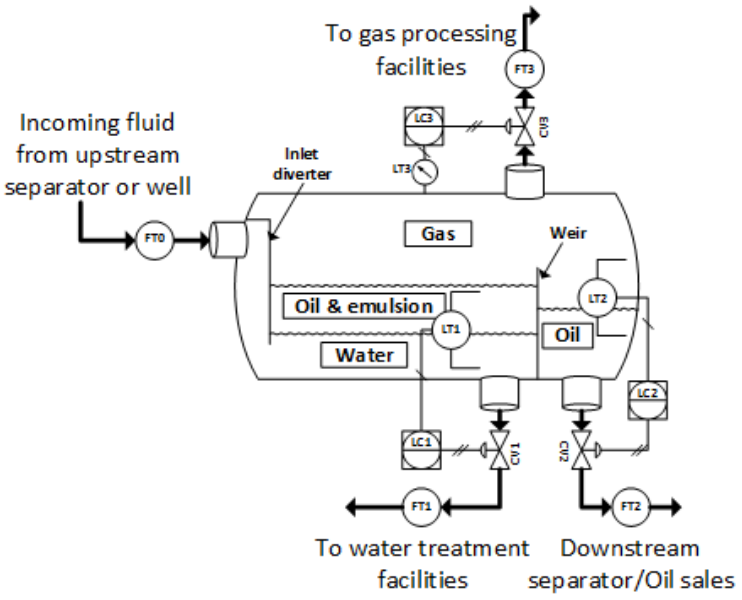


Figure 3.1. Horizontal three-phase separator schematic of configuration with weir

The process of oil, gas and water separation can be briefly described as follows (Arnold and Stewart, 2008):

1. The first separation occurs when the incoming fluid hits an inlet diverter and the heavier oil and water emulsion settles on the bottom of the vessel (the gravity settling section). The lighter gas concentrates at the top of the vessel (the separated gas section) and leaves the vessel through the gas outlet.
2. In the gravity settling section the water separates from oil. The more dense water goes to the bottom of the vessel, while the oil remains on top of the water. The water then leaves the vessel through the outlet at the base of the vessel.
3. The oil that goes over the weir (in the separated oil section) leaves the vessel through an oil outlet.

The main internal components of the horizontal three-phase separator are listed in Table 3.1.

Table 3.1. Internal components of the three-phase separator (Arnold and Stewart, 2008)

Component	**Description**
Inlet diverter	An inlet diverter is usually a metal plate or a spherical dish, which performs the very first separation of the incoming mixture of fluids. The inlet diverter provides a rapid change in the velocity and direction of the incoming mixture. This sudden change makes the heavier liquids (water and oil) flow below the diverter to the bottom of the vessel while the lower density gas flows above the diverter towards the top of the vessel.
Weir	A separated oil section of the vessel is created by using a weir. The weir physically blocks the passage of a heavier liquid to the next section of the vessel. Since the oil floats on top of the water (due to lower density), it is the only liquid flowing over the weir to the separated oil section, if the water level is kept at a desired level below the weir. The level of water (the interface level between water and oil) should be controlled so that it would never flow over the weir. Otherwise the water would mix with the oil, which has already separated.

For the efficient and safe operation of the three-phase separator, an automatic control mechanism has to be set up to control the water and oil interface level in the gravity settling section, the separated oil level in the separated oil section and the pressure in the vessel. The components used to monitor and control the three-phase separator, summarised in Table 3.2, can be grouped based on the process variable that they monitor and control.

The water level is monitored and controlled by components LT1, LC1, CV1 and FT1, the oil level is monitored and controlled by components LT2, LC2, CV2 and FT2 and the pressure level – by components LT3, LC3, CV3 and FT3. Each group of components is known as a closed control loop, which is discussed in more detail in Section 3.1.4.2. The common control loop feature can be exploited when developing a generic methodology for the fault detection and identification of systems similar to that of the three-phase separator, such as oil desalters or produced water treatment systems. A generic fault detection and identification module can be created for a control loop.

Table 3.2. Components to monitor and control three-phase separator (Arnold and Stewart, 2008)

Name	Component	Purpose
FT0	Flow rate transmitter	Measures the multiphase flow (water, oil and gas individually) coming to the separator from the upstream equipment. Provides the information to predict the water, oil and gas level changes in the separator.
LT1	Level transmitter	Measures the levels of water and oil in the gravity settling section. Provides the information to control the water-oil interface level for safe and efficient use of the separator.
LC1	PI controller	Provides a control command to keep control valve CV1 at the necessary opening so that the water-oil interface is maintained at a desired level (based on Proportional Integral (PI) control mode further described in section 3.1.4). Receives the information from the LT1 level transmitter.
CV1	Control valve	Provides a way to control the water-oil interface level in the gravity settling section by opening/closing when the corresponding command is received from the controller LC1.
FT1	Flow rate transmitter	Measures the flow of the liquid allowed by the opening of the control valve CV1 to monitor the outflowing water from the separator.
LT2	Level transmitter	Measures the level of oil in the separated oil section. Provides the information to control the oil level for safe and efficient use of the separator.
LC2	PI controller	Provides a control command to keep control valve CV2 at the necessary opening so that the separated oil is maintained at a desired level. Receives the information from the LT2 level transmitter.
CV2	Control valve	Provides a way to control the oil level in the separated oil section by opening/closing when the corresponding command is received from the controller LC2.
FT2	Flow rate transmitter	Measures the flow of the liquid allowed by the opening of the control valve CV2 to monitor the outflowing separated oil from the separator.
LT3	Pressure transmitter	Measures the pressure in the separator. Provides the information to control the pressure inside the vessel for safe and efficient use of the separator.
LC3	PI controller	Provides a control command to keep control valve CV3 at the necessary opening so that the pressure is maintained at a desired level. Receives the information from the LT3 level transmitter.
CV3	Control valve	Provides a way to control the pressure inside the separator by opening/closing when the corresponding command is received from the controller LC3.
FT3	Flow rate transmitter	Measures the flow of the gas allowed by the opening of the control valve CV3 to monitor the outflowing gas from the separator.

3.1.2 Undesirable events in the water, oil and gas separation

There are six undesirable events that can occur during three-phase separation, as identified by API 14C (American Petroleum Institute, 2001): overpressure, under-pressure, liquid overflow (or liquid carryover), gas blowby, leak and excess temperature if the vessel is heated. In this study, the four first events are considered.

The undesirable events have a potential to cause injury, pollution, or damage, and this fact has to be taken into account when designing a protection system to prevent such events. Usually, two levels of protection are implemented in the separators. Primary protection must be provided either to prevent the undesirable event from occurring or to minimise its effects. Secondary protection must be provided as a backup if the primary protection fails. These levels of protection can be provided by individual devices (e.g., pressure safety high/low sensors, level safety high/low sensors) or by specially designed systems.

The four undesirable events that are relevant to the three phase separator considered in the study are now described in more detail. An overpressure is a pressure which exceeds the maximum allowable working pressure. An overpressure of the vessel might lead to a sudden rupture of the vessel or piping and can cause the leak of hydrocarbons. The main causes of overpressure include a pressure gas valve failed closed, a situation when the inflowing amount of gas significantly exceeds the outflowing amount, gas blowby in the upstream component or the failure of a pressure control system. The primary protection from overpressure is a high pressure safety sensor, whose readings warn about a need to shut off the inflow. The secondary protection is provided by a pressure safety valve, which should be opened if the pressure is not reduced by the cutoff of inflow.

An underpressure is defined as a pressure which is lower than the designed collapse pressure. This is a hazardous event, since the vessel can collapse and release hydrocarbons into the environment. It might also be followed by explosions or fires. The main cause of underpressure is a significantly larger amount of gas outflowing the vessel than that inflowing the vessel. Such a situation might occur if the inlet valve is blocked and the outlet valve fails open. The primary protection from underpressure is provided by a gas makeup system, i.e. gas is added to the vessel through an alternative inlet. The secondary protection is provided by a low pressure safety sensor to initiate the shut off of the inflow and outflow.

A liquid overflow is a process when liquids are discharged from the vessel through the gas outlet. The effects of the liquid overflow can be either excess liquid in a downstream component, overpressure or release of hydrocarbons into the atmosphere. It is usually a consequence of an incoming flow rate exceeding the limit that the vessel has been designed for, improper water or oil level control or instantaneous liquid inrush. The primary protection from liquid overflow is achieved by initiating the shut off of the inflow based on the readings from a high level safety sensor. The secondary protection is provided by a standalone Emergency Support System.

One of the most hazardous situations that might occur during the operation of a three-phase separator is a gas blowby. Gas blowby is a process when gas leaves the vessel through the oil or water outlet and thus causes an over-pressure in the downstream vessel or pipeline. The over-pressure of the vessel or pipeline might lead to an explosion, fire and further hazardous events in other parts of the plant. Gas blowby might occur if the liquid level in the separator is low, which is usually caused by a failure of the liquid level control system, or if vortices form near the

outlets, which suck in gas together with liquid. The primary protection for gas blowby is a low liquid level sensor that initiates the shut off of the outflow from the vessel when the liquid level drops below the lowest acceptable operating level. The secondary protection is installed in the downstream process components, which is equipped with a pressure safety sensor and a valve to protect them from the consequences of a gas blowby.

3.1.3 Failure mode and effects analysis

A simple, yet effective technique to identify possible failure scenarios of a system is Failure Mode and Effects Analysis (FMEA). The idea is to list components of the system by considering how system performance would be affected if that particular component failed in a particular failure mode. It is a good technique to identify the system weaknesses at a component level. The downside of FMEA is that it only considers single failures of components and does not take into account combinations of failures, which are of more concern in most industrial systems.

A simple FMEA was performed for the three-phase separator as a part of the study and the results are presented in Table 3.3. Note that the FMEA was performed for a set of selected failure modes and the list of failure modes considered is not exhaustive. The FMEA was performed to get a better understanding how single failures can influence the control and performance of the three-phase separator. The findings were used when developing the simulator model and when building the OOBN model for the fault detection and diagnostics.

Table 3.3. FMEA of three-phase separator

Component name	Function	Failure mode	Failure effect	Failure detection
Flow rate transmitter FT0	Measures the flow rate of incoming mixture	Failed stuck	**Local effect:** The flow rate measurement value remains constant **System effect:** No effect on system performance, since it is used only for monitoring purposes	The levels of water and oil (indicated by LT1 and LT2) and pressure (indicated by LT3) are changing differently than expected given flow rates indicated by FT0.
Level transmitter LT1	Measures the levels of water and total liquid in the gravity settling section	Failed stuck	**Local effect:** The level measurement value remains constant **System effect:** Improper control of the water level that could lead to gas blowby or liquid carryover	The water level shown as constant by LT1, when it is expected to change, constant water outflow (indicated by FT1) when the water inflow changes (indicated by FT0), total liquid level shown as constant.
PI controller LC1	Send the command to the control valve CV1	Failed high	**Local effect:** Sends the command to open the CV1 no matter what the water level is in the vessel **System effect:** Low water level that could lead to gas blowby or liquid carryover	A high outflow rate in FT1 even when the CV1 is supposed to be closed based on LT1 readings
		Failed low	**Local effect:** Sends the command to close the CV1 no matter what the water level is in the vessel **System effect:** High water level that could lead to liquid carryover	No outflow in FT1 even when the CV1 is supposed to be opened based on LT1 readings

Component name	Function	Failure mode	Failure effect	Failure detection
Control valve CV1	Opens or closes the way for leaving water	Failed open	**Local effect:** Uncontrolled water outflow **System effect:** Low water level that could lead to gas blowby	A high outflow rate in FT1 even when the outlet valve CV1 is supposed to be closed based on LT1 readings
		Failed closed	**Local effect:** No water outflow **System effect:** High water level that could lead to liquid carryover	No outflow in FT1, even when the outlet valve is CV1 supposed to be opened based on LT1 readings
Flow rate transmitter FT1	Measures the flow rate of leaving water	Failed stuck	**Local effect:** The flow rate measurement value remains constant **System effect:** No effect on system performance, since it is used only for monitoring purposes	The level of the water - oil interface (indicated by LT1) is changing differently than expected with a given flow rate (indicated by FT1).
Level transmitter LT2	Measures the separated oil level	Failed stuck	**Local effect:** The level measurement value remains constant **System effect:** Improper control of oil level that could lead to gas blowby or liquid carryover	Oil level shown as constant by LT2, when it is expected to change, constant oil outflow (indicated by FT2) when the oil inflow changes (indicated by FT0) and the oil level is equal with the weir in the gravity settling section (indicated by LT1).
PI controller LC2	Sends the command to the control valve CV2	Failed high	**Local effect:** Sends the command to open the CV2 no matter what oil level is in the vessel **System effect:** Low oil level that leads to gas blowby	A high outflow rate in FT2 even when the CV2 is supposed to be closed based on LT2 readings
		Failed low	**Local effect:** Sends the command to close the CV2 no matter what oil level is in the vessel **System effect:** High oil level that could lead to liquid carryover	No outflow in FT2 even when the CV2 is supposed to be opened based on LT2 readings

Component name	Function	Failure mode	Failure effect	Failure detection
Control valve CV2	Opens or closes the way for leaving oil	Failed open	**Local effect:** Uncontrolled oil outflow **System effect:** Low oil level that could lead to gas blowby	A high outflow rate in FT2 even when the outlet valve CV2 is supposed to be closed based on LT2 readings
		Failed closed	**Local effect:** No oil outflow **System effect:** High oil level that could lead to liquid carryover	No outflow in FT2, even when the outlet valve is CV2 supposed to be opened based on LT2 readings
Flow rate transmitter FT2	Measures the flow rate of leaving oil	Failed stuck	**Local effect:** The flow rate measurement value remains constant **System effect:** No effect on system performance, since it is used only for monitoring purposes	The level of oil (indicated by LT2) is changing differently than expected with a given flow rate (indicated by FT2).
Pressure transmitter LT3	Measures the pressure in the vessel	Failed stuck	**Local effect:** The pressure measurement value remains constant **System effect:** Improper control of pressure level that could lead to under-pressure or over-pressure	Pressure shown as constant by LT3, when it is expected to change, constant gas outflow (indicated by FT3) when the gas inflow changes (indicated by FT0)
PI controller LC3	Sends the command to the control valve CV3	Failed high	**Local effect:** Sends the command to open the CV3 no matter what pressure is in the vessel **System effect:** Low pressure that could lead to under-pressure	A high outflow rate in FT3 even when the CV3 is supposed to be closed based on LT3 readings
		Failed low	**Local effect:** Sends the command to close the CV3 no matter what pressure is in the vessel **System effect:** High pressure that could lead to over-pressure	No outflow in FT3 even when the CV3 is supposed to be opened based on LT3 readings

Component name	Function	Failure mode	Failure effect	Failure detection
Control valve CV3	Opens or closes the way for leaving gas	Failed open	**Local effect:** Uncontrolled gas outflow **System effect:** Low pressure that could lead to under-pressure	A high outflow rate in FT3 even when the outlet valve CV3 is supposed to be closed based on LT3 readings
		Failed closed	**Local effect:** No gas outflow **System effect:** High pressure that could lead to over-pressure	No outflow in FT3, even when the outlet valve is CV3 supposed to be opened based on LT3 readings
Flow rate transmitter FT3	Measures the flow rate of leaving gas	Failed stuck	**Local effect:** The flow rate measurement value remains constant **System effect:** No effect on system performance, since it is used only for monitoring purposes	The pressure inside the separator (indicated by LT3) is changing differently than expected with given volume available to the gas and flow rate (indicated by FT3).

3.1.4 Automatic control of three-phase separators

Since most processes in the oil and gas industry include hazardous materials, it is very important that a process remains within safe operable limits. The control of the process is achieved by keeping specific parameters of the process at a desired level, usually defined as a set point (SP). Usually the parameters are controlled automatically. However, the supervision of the automatic control is also required.

An effective automatic process control is especially important for the three-phase separators, since it not only guarantees economic benefits, but more importantly, it also guarantees the safe operation of the separator. A temperature, liquid level, flow and pressure are commonly controlled during the gas and oil processing.

The majority of controllers found in industrial applications are Proportional Integral (PI) controllers with over 90% of total population and these are the ones that are used in the control of three-phase separators (Svrcek et al., 2006). A control function provided by this type of controller is based on the PI algorithm, which is presented next.

3.1.4.1 Proportional Integral controllers

The PI control algorithm is based on two actions: change the controller output **proportionally** to the error (deviation from a set point) and **integrate** previous errors of the controller.

Most controller algorithms these days are implemented using digital electronics, thus a discrete form of a PI controller output has to be used. A discrete form of the PI algorithm can be written in two different ways: a positional or a velocity form. The positional form of the controller algorithm indicates the current position of the final control element (FCE), e.g. a valve, while the velocity form indicates in which direction and how much the position of the FCE has to be changed. The positional form of the discrete PI control algorithm output is:

$$CO(n) = CO_b + K_C\left(e(n) + \frac{\Delta t}{T_i}\sum_{k=1}^{n} e(k)\right), \qquad \textbf{3.1}$$

where n is the discrete sampling time point, $CO(n)$ is the controller output at time n, CO_b is the controller output bias value, K_C is the controller gain, $e(n) = SP - MV(n)$ is the error at time n, SP is the set point or desired value of the controlled variable, $MV(n)$ is the measured value of the controlled variable at time n, Δt is the time difference between two sampling times, T_i is the integral time.

However, this form of the algorithm suffers from a phenomenon known as a reset or integral windup. This is a phenomenon when the integral sum $\sum_{k=1}^{n} e(k)$ in equation **3.1** grows to a limit that the control output $CO(n)$ requires the FCE to open or close more than it is physically capable (FCE saturates). For example, a liquid outlet valve is fully opened, but the liquid inflow is greater than the maximum allowable outflow. The integral sum will keep increasing and the controller will keep the outlet valve fully opened for longer than it is necessary when the inflow decreases. If a control algorithm is not properly secured from such a phenomenon, control actions might be inadequate and result in extreme points of the operation (i.e. draining the tank or overfilling it).

There are two ways to avoid integral windup. One way is to use some bounding logic to avoid the sum becoming too large. The other way is to use a velocity form of the PI control algorithm. The velocity form of the PI control algorithm can be defined as the difference between two control outputs:

$$\Delta CO(n) = CO(n) - CO(n-1) \qquad \textbf{3.2}$$

The velocity form of the PI control can be obtained by using equation **3.1** and equation **3.2**:

$$\Delta CO(n) = K_C \left(\left(1 + \frac{\Delta t}{T_i}\right) e(n) - e(n-1) \right), \qquad \textbf{3.3}$$

where $e(n)$ is the error at sampling time n, $e(n-1)$ is the error at a previous sampling time $n-1$, Δt is the time difference between sampling points n and $n-1$.

There are several ways in which the velocity form algorithm is used. One way is to use it as it is given in equation **3.3**, when only a change in the command is sent to the FCE. This way the FCE itself provides protection from integral windup: when the FCE is at its maximum or minimum opening, the command to open or close it more will not have effect since it cannot be physically opened or closed more. The other way is to avoid sending a command which would need the FCE to be moved above its physical limits, as suggested in (Smith, 2009):

1. Calculate a change in command $\Delta CO(n)$ as in equation **3.3**.

2. Check if such a change in the FCE position would force the FCE above its physical limits and adjust $\Delta CO(n)$ correspondingly:

$$\Delta CO(n) = \begin{cases} \Delta CO(n), & if\ CV_{Min} \le CO(n-1) + \Delta CO(n) \le CV_{Max} \\ 0, & otherwise \end{cases} \qquad \textbf{3.4}$$

3. Calculate the command sent to the FCE as following:

$$CO(n) = CO(n-1) + \Delta CO(n)\,, \qquad \textbf{3.5}$$

The algorithm defined by the three steps above will be implemented to model the velocity form of the PI control algorithm in the three-phase simulation model in this study to avoid the undesirable effects caused by the integral windup phenomenon.

3.1.4.2 Control loops in the three-phase separator

The control of the separation process is achieved by controlling three key process variables: a water-oil interface level in the gravity settling section, an oil level in the separated oil section and a pressure in the separator. Each key process variable in the three-phase separator has its own control loop (see Figure 3.2): a water-oil interface level control loop, a separated oil level control loop and a vessel pressure control loop.

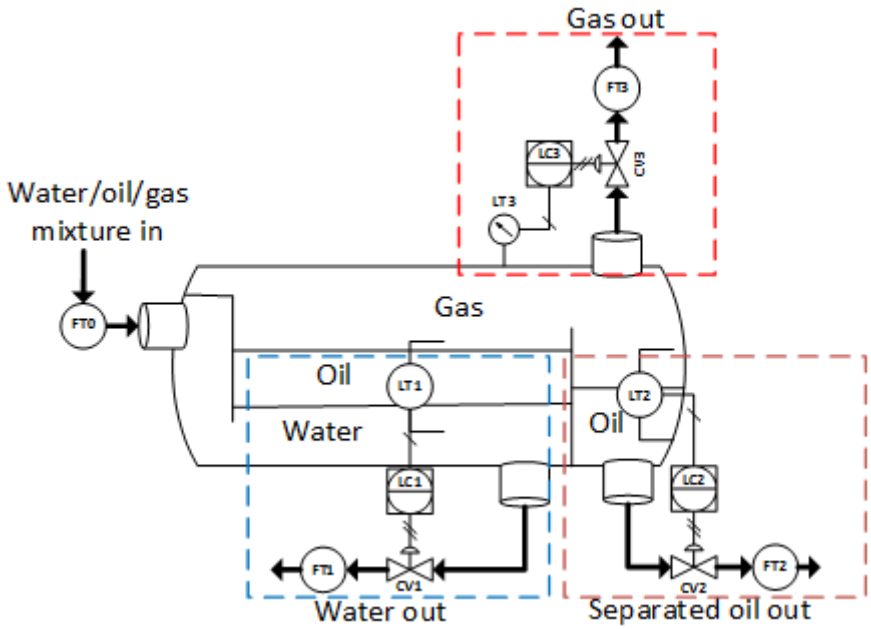


Figure 3.2. Control loops of a three-phase separator: water-oil interface level (represented by blue rectangle), separated oil level (represented by brown rectangle) and pressure level (represented by red rectangle) control loops

In this study, all of the control loops, as mentioned in section 3.1.1, are controlled by PI controllers.

Interface level control

The water-oil level control loop consists of transmitter LT1 (for measuring the interface level), PI controller LC1 and valve CV1 as FCE. Flow transmitter FT1 is not used in the control loop, but it indicates what flow is allowed through valve CV1 and is important for detecting abnormal conditions of water flow from the vessel.

An interface level between oil and water has to be maintained at a desired set point for normal operation of the separator. If the interface level is high, less space is available for the oil separation and thus the quality of separated oil might be affected. Furthermore, if the interface level control fails, there is a possibility for water to flow over the weir. This would be detrimental to the whole separation process, since the water would mix with the separated oil, which is kept in the adjacent compartment of the vessel. If the interface level is low, the separation process is faster than the oil transfer into the separated oil section and thus the separator is not working efficiently. Moreover, if the interface level is very low, some oil might be outflowing from the vessel through water drain valve CV1. This would result in economic losses. Two worst case scenarios might occur due to a poor interface level control: the vessel might become overfilled, leading to overpressure, liquid overflow or leaking, or the vessel might become drained causing underpressure and gas blowby, as described in section 3.1.2. Such situations result in risk to personnel and economic losses.

Separated oil level control

The control of the separated oil level is simple, since there is only one liquid in the separated oil section. A simple float device might be sufficient to measure the level of oil. As in the interface level control, there is transmitter LT2, which measures the oil level and sends the information to the PI controller LC2, which then sends the control command to valve CV2.

If the control of the oil level is lost and there is no outflowing oil through valve CV2, the level might rise up to a point where it starts flowing back over the weir into the gravity settling section. This would interfere with the separation process and would reduce the efficiency of the separator.

If the control of the separated oil level fails, an undesirable event, i.e. gas blowby, might occur. If the oil level becomes very low, there is a chance that some gas might be sucked in through the valve CV2 together with the outflowing oil. If necessary safety systems are not installed or they have failed, this may lead to a very hazardous situation, since only oil is expected to leave through CV2 and the downstream equipment is usually set up to operate at a lower pressure. When there is a significantly large amount of gas still left in the oil, the downstream equipment might be overpressured and eventually rupture, releasing hydrocarbons into the processing plant and the environment. Hydrocarbon release is a potential source of fire, if there is a possibility of a spark happening nearby.

Thus, the effective control of the oil level is not only important for the efficiency of the process but also in terms of safe operation of the three-phase separator.

Vessel pressure control

The control of pressure in the vessel is somewhat different from liquid level control. Since gas can be compressed, pressure in the vessel not only depends on the molecular amount of gas, but also on the volume available for gas and the temperature in the vessel. Thus, the same volume of gas might result in a different pressure in the vessel. In this study, it is considered that the vessel is not heated and the temperature inside the vessel is constant. It is also assumed that the ideal gas law holds:

$$PV = nRT, \tag{3.6}$$

where P is the pressure of gas in pascals (Pa), V is the volume of gas in cubic meters (m^3), n is the amount of gas in moles (mol), R is the Avogadro constant ($JK^{-1}mol^{-1}$), T is the temperature in Kelvins (K).

Thus the pressure in the vessel can be expressed from equation **3.6** as:

$$P = \frac{nRT}{V}. \tag{3.7}$$

As R is a constant and T is assumed to be constant, the volume of gas V and the amount of gas n are the variables that change and determine the pressure P.

The pressure control loop consists of the transmitter LT3, PI controller LC3 and valve CV3. As in the previous two control loops, flow transmitter FT3 is not used for control purposes, but it indicates the gas flow allowed through valve CV3 and is important for monitoring the gas outflow. The pressure control loop can have two failure modes: overpressure or underpressure. Both of these failure modes are considered as undesirable events and were described in section 3.1.2.

3.1.5 Summary

A three-phase oil, water and gas separator was presented in this section. A description of the system was given, identifying its main components and their functionality. The components used in three-phase separators were grouped into

control loops, providing a potential to produce a generic fault detection and diagnostic methodology. Common undesirable events were presented next, as identified by the American Petroleum Institute. A brief description of each undesirable event was given, identifying their causes and consequences. Common failure modes of the considered system and their effects were analysed using FMEA. The FMEA allowed a better understanding of the system and how single component faults influence the performance of the separator. This will aid the building of the simulation model of the separator as well as the developing the fault detection and diagnostic methodology.

The topic of control loops was revisited by presenting a theory of automatic process control by highlighting the chosen control method with PI controllers used in the study. This control method uses a velocity form of the PI control algorithm in order to overcome the issues of windup. The control loops used in a three-phase separator introduced earlier were described in more detail emphasising how improper control in a specific loop can result in economic losses or hazardous situations.

A summary of findings from a literature review of the fault detection and diagnostics techniques applied in the oil and gas industry is given in the next section.

3.2 Literature review

The literature review section is divided into two major subsections: firstly the methods for fault detection and diagnostics in the oil and gas industry are presented, followed by a more detailed description of Bayesian Belief Networks.

3.2.1 Fault detection and diagnostics in oil and gas industry

An extensive literature review was performed to identify the state of the art of methods used for fault detection and diagnostics in the oil and gas industry. The reviewed approaches are divided into three categories: model based approaches, expert systems and statistical analysis approaches. These methods are discussed in the next subsections by introducing the problem they solve, the methodology that was proposed and how the methods were tested and validated.

3.2.1.1 Model based approaches

Diagnostic Model Processor

Fault diagnostics of an oil production plant prototype was performed using a Diagnostic Model Processor in (Dias et al., 1993). The main idea of the Diagnostic Model Processor is to evaluate a set of residuals $E = (e_1, e_2, \dots e_N)$, generated by equations, which define the condition of the system at current time and compare them to residuals generated by the steady state of the system under a set of assumptions $A = (a_1, a_2, \dots, a_M)$, which represent the fault-free condition of the system, e.g. $a_1 =$ "There is no bias in the output signal of level controller LIC-1". The comparison is done by calculating the likelihood of the violation of the assumptions:

$$P_i = \frac{\sum_{j=1}^{N}(S_{ij} sf_j)}{\sum_{j=1}^{N}|S_{ij}|}, \qquad \textbf{3.8}$$

where P_i is the likelihood of violating assumption a_i, S_{ij} is the sensitivity of residual e_j with respect to assumption a_i, sf_j is the violation level that indicates the degree to which residual e_j is satisfied.

The drift of the residuals from zero values would indicate that the system has deviated from the steady state due to a failure. The increasing likelihood of the violation of the assumptions would allow a distinct failure of the system to be isolated. If several failures have an increase in likelihood, the ones with $P_i > 0.5$ are favoured.

The measurements of oil level, pressure, oil temperature, hot water temperature, and oil outflow sensors were used for the analysis. The Diagnostic Model Processor contained 38 residual equations with 51 assumptions. Results with several failure modes were illustrated. In some cases the model could correctly identify the violation of some assumptions, such as "There is no bias in the pressure controller input PIC-1" but failed to indicate the violation of the assumption "Loss of air supply to the pneumatic mechanism of control valve LCV-2".

The authors have concluded that their model is capable of identifying single failures. However, in the majority of the cases the model finds more than one failure even when only a single failure occurs. Moreover, the authors indicated that a very good understanding of the system processes is needed in order to define the steady-state equations for the oil production plant and the steady-state cannot change with time since that would require a set of new residual equations.

Linear model and Kalman filter

Another model-based approach for fault detection in process control systems was proposed in (Afonso et al., 1998). The authors used a stirred tank prototype to implement and test their approach. The whole system consisted of three tanks (cold water tank, hot water tank and stirring reactor tank) and a cooling system. Water from cold and hot water tanks was flowing into a stirring tank, which was partially cooled by a cooling system. The process in the stirred tank was defined using a linear model given by equations **3.9** and **3.10**:

$$h_{k+1} = h_k + \frac{\Delta t}{A}\left(Q_{1,k} + Q_{2,k} - Q_{3,k}\right), \quad \textbf{3.9}$$

$$T_{k+1} = T_k - \frac{\Delta t U_g \pi dr}{\rho C_p A}\left(T_k - Tc_k\right) + \frac{\Delta t}{A}\left(\frac{Q_{2,k}}{h_k}\left(T_{2,k} - T_k\right) + \frac{Q_{1,k}}{h_k}\left(T_{1,k} - T_k\right)\right) + \frac{\Delta t U_g}{\rho C_p h_k}\left(Tc_k - T_k\right), \quad \textbf{3.10}$$

where $h_{k+1},\ h_k$ are the heights of liquid in the stirring tank at time steps $k+1$ and k respectively, $\Delta t = 0.1\ min$ is the size of the time step, $Q_{1,k},\ Q_{2,k},\ Q_{3,k}$ are cold water inflow, hot water inflow and mixed liquid outflow rates at time step k respectively, $T_{k+1},\ T_k$ are the temperatures of liquid in the stirred tank at time steps $k+1$ and k respectively, Tc_k is the temperature of coolant at time step k, $T_{1,k}$, $T_{2,k}$ are temperatures of inflowing cold and hot water respectively at time step k, and operating conditions $U_g = 962°Jdm^{-2}min^{-1}K^{-1}$, $A = 17.6\ dm^2$, $dr = 4.73$, $C_p = 4148JKg^{-1}K^{-1}$, $\rho = 1Kgdm^{-3}$. The model parameters in equations **3.9** and **3.10** were estimated with a Recursive Least Squares algorithm at each time step, when observations from the stirring tank were obtained.

The fault detection stage involved performing statistical tests on the model parameters in equations **3.9** and **3.10**. The model parameters were assumed to follow a Normal distribution in the selected time window, and the Fisher statistic was used to check the hypothesis of equal variances of two subsequent time windows. The width of the time window was found experimentally to be equal to 101 time steps. If the variances of model parameters of the subsequent time windows were found to be unequal, an alarm was issued.

When a failure was detected, an Extended Kalman Filter was used to estimate the state of the process and compare it to sensor readings to identify the failure cause. If the estimated state was different from the sensor readings, the sensors were considered to be faulty. The authors, however, did not provide details of how the estimated state and sensor readings were compared. Moreover, the assumption was made that multiple failures could not occur simultaneously. The authors did not provide the exact figures for the efficiency of the method, and only stated that the proposed technique gave promising results and further research had to be performed in order to improve the technique.

CUSUM test on modelled system residuals

A fault detection and isolation technique based on residual patterns of a gas-liquid separator (system of two connected tanks for separating gas from water) model was presented in (Kinnaert et al., 2000). Five measurements from the system (pressure, water level in the first tank, water level in the second tank, flow of air from the first tank and flow of water from the second tank) were available, thus providing five residual signals. The fault detection and isolation methodology proposed was based on the fact that fault-free residual signals should follow a Normal distribution with a

zero mean, while faulty residual signals should follow a Normal distribution with a non-zero mean. The variances of these signals were assumed to be equal.

A cumulative sum control chart (CUSUM) test was performed on the residuals to test whether they had a fault-free or faulty distribution with selected threshold values for fault detection and isolation. The proposed approach was tested with a step-like bias introduced to the five readings mentioned above with five different magnitudes. The results presented showed that the performance of the fault detection and isolation technique highly varied for different fault magnitudes and different fault detection thresholds. The authors indicated that the technique needed further testing in the presence of modelling uncertainties and validation with the actual plant data.

Petri Net model

The research presented in (Al-Hajri and Rossiter, 2010) developed a fault detection and isolation technique based on a Petri Net model. A methodology for building Petri Nets for an oil producing station was presented. The whole station model was built by joining smaller Petri Nets developed for individual components of the oil producing station. Fault detection and isolation was performed by comparing two Petri Nets: a Petri Net which represented the normal behaviour of the system (places of the Petri Net represented the components with working states only) and a diagnostic Petri Net (places of the Petri Net involved failed states of the components). Whenever the markings of places in these two Petri Nets were different, the system was considered to be faulty. The proposed approach was validated using an example of a flow control valve unit. It was shown that the model built was able to detect failures of pressure and flow transmitters in the flow control valve unit. The authors indicated that their proposed approach could decrease the gap between academic research in fault detection and diagnostic techniques and their application in industrial solutions, but validation of the Petri Net technique should be performed on a real system.

3.2.1.2 Expert systems

An expert decision support system for petroleum production and separation processes was presented in (Chan, 2005). The main goal of this research was to acquire knowledge about the petroleum production and separation process from a number of engineers, so that common information could be obtained for an expert decision support system. From the knowledge obtained, if-then rules were formed to give warnings to operators, when certain parameters of the process deviate from normal operating conditions. Additionally, other control actions were suggested by the decision system if knowledge on how to control the deviating process was available. The decision system was evaluated by two expert engineers, who highlighted that the system outputs were useful and valid for the chosen malfunctioning scenarios. Specific details about the rules of the decision support system were omitted in this paper and only an example of rules for a water treater temperature assessment was given:

- If temperature is 125-135 F, then no action.
- If temperature is 100-125 F or 135-150F, then a yellow warning is given, no action.
- If temperature is 65-100 F or 150-185 F, then a red warning is given, no action.
- If temperature is 185-200 F, an alarm is given and suggestion to operators is given: “this might be caused by (1) no exit flow from treater, (2) heater

malfunction", and actions suggested to the operator are: (1) turn off the heater, (2) if (1) does not work, then shut down production pump.

3.2.1.3 Statistical analysis approaches

Autonomous recursive task decomposition

A statistical analysis approach to diagnose faults on industrial plants was suggested in (Roverso, 2002). Roverso proposed a methodology that can be applied to any industrial plant, given measurements of process variables are available and they change as a failure occurs. The proposed methodology consisted of recurrent neural network ensembles, a wavelet on-line pre-processing (WOLP) and autonomous recursive task decomposition (ARTD).

First of all measurements of the process variables were pre-processed with the WOLP technique to reduce a dimensionality of the input data, extracting the mean residual as well as the maximum and minimum wavelet coefficients from the input data. Since the plants usually consist of many components, there were many measurements available which could be used for the diagnostics of the plant. Moreover, a lot of different failure modes might occur in the plant. This led to a problem when a neural network had to be trained to classify a large amount of data into a lot of different classes (32 classes in this case). Thus the ARTD method was presented to decompose the classification problem into smaller sub-problems. The results of neural networks from sub-problems were then combined into a tree, forming an ensemble of recurrent neural networks, which was then used for the plant diagnostics.

The proposed methodology was tested with artificial data representing dynamics of the plant process. The author showed that decomposition of the classification into 32 classes to smaller sub-classifiers led to an increase in algorithm speed allowing similar or even better results to be obtained than those without the decomposition. The downside of the proposed methodology is that it requires intensive computing resources for network training with a large amount of data representing all the possible dynamics of the process under consideration. This data might not always be available, especially for failure scenarios.

Generalised Parity Vector

The fault detection and isolation of a three-phase separator was performed using a Generalized Parity Vector technique in (Omana and Taylor, 2007) and (Taylor and Omana, 2008). The idea of the technique is to form vectors from system input and output variables (e.g. incoming oil stream and outcoming oil stream) with a big angle between them when the system is in a non-faulty state. Whenever a big difference between system input and output occurs, the angle between these vectors reduces to almost zero, this way indicating a failure in the system. Different failures should produce parity vectors with different characteristics and that would be the way to isolate the failures.

The authors tested the technique with several scenarios generated by a simulated model of the three-phase separator, showing that the fault detection is straightforward and their technique is capable of detecting failures as soon as they are inserted in the simulation model. However, the identification of failures was more difficult and several angles of parity vectors changed, indicating several different failure modes even when a single failure was simulated. The technique proposed by

the researchers was implemented in a system to monitor, control and manage the petroleum production facilities model in (Taylor and Sayda, 2008). The same deficiency was observed as the proposed technique was not able to isolate failures.

Principal Component Analysis

A methodology combining Principal Component Analysis (PCA) and wavelet analysis was proposed to diagnose faults for an oil gas water separator in (Gao et al., 2009). An idea of the methodology was to pre-process the data obtained from the simulation model of oil gas water separator with a wavelet analysis and use the pre-processed data to find the principal components. Hotelling T^2 and Squared Prediction Error (SPE) statistics were calculated from the principal components. Thresholds for these statistics were set experimentally. Whenever these thresholds were breached, it was considered that a failure had occurred in the system.

Several scenarios were tested by the authors to validate the performance of the methodology. One scenario showed a clear identification of a failure in the system, while another showed that the calculated statistics were fluctuating above and below the set thresholds. Thus the fault detection was dependable on a good selection of thresholds for the Hotelling T^2 and SPE statistics, which might not be straightforward in all cases.

A fault detection and identification technique based on PCA for monitoring the condition of an offshore oil and gas production process was also presented in (Natarajan and Srinivasan, 2010). A dynamic model of an offshore system was first built to obtain performance data, such as measurements of flow rates of water or oil, for the system in fault-free, faulty and under-maintenance states. In total, 32 different measurements were available from the dynamic model. These measurements were clustered using a k-means algorithm into 13 clusters which were then used to build PCA models for normal operation and for operation with faults present. Whenever new data became available it was fitted to each of 13 previously obtained clusters and the PCA model corresponding to the best-fit cluster was chosen. For the chosen PCA model, the Q statistic was calculated. If the Q statistic was within the confidence limits for the selected PCA model, the process was considered to be fault-free. Alternatively, if the Q statistic exceeded predetermined confidence limits, the process was considered to be faulty.

Three different fault scenarios to test the proposed methodology were presented in this paper: a valve leak fault, a test separator temperature controller fault and a temperature sensor fault. The testing performed showed that all three scenarios were identified early, i.e. before the failures affected the process to a large extent. Even though the proposed methodology was shown to be useful for fault detection and identification, an individual PCA model had to be built for each failure mode considered. This is a significant deficiency when a large system with a high number of failure modes is considered. Moreover, multiple failure modes might occur in the system and thus individual PCA models would need to be built for all possible failure combinations.

3.2.1.4 Summary

A variety of techniques have been developed for the detection and diagnostics of faults in equipment used in the oil and gas industry. Three broad categories were identified: model-based approaches, expert systems and statistical analysis approaches.

Model-based approaches needed a very good understanding of the processes considered in order to be able to build a precise mathematical model for the system. Testing of the developed model residuals was one of the common ways to perform the fault detection and diagnostics (Dias et al., 1993, Kinnaert et al., 2000). With such a technique, subtle changes in the residuals generated by the model can be identified and in some cases a certain failure mode can be isolated. However, if the normal process operating conditions change, for example inflows of liquids or a set point of liquid levels change, the whole mathematical model has to be rebuilt to take into account those changes. Moreover, extensive testing of the proposed approaches was not reported in the papers, thus it is hard to evaluate the actual performance of the proposed methodology. The detail of results presented in (Afonso et al., 1998) again does not allow evaluation of the actual performance of the methodology. The authors also indicated that their technique is invalid for multiple failures. This is a big deficiency of the method, since it is common that multiple failures occur on real systems. A Petri Net based approach presented in (Al-Hajri and Rossiter, 2010) was another example of a fault detection technique from this category. The authors indicated that the initial results were promising; however as in previously described examples, extensive testing with different failure modes was not presented in the study. Therefore, clearer conclusions cannot be drawn.

A simple expert system based on ‘if-else’ conditions was presented in (Chan, 2005). The main deficiency of such an approach is that it cannot take into account complex relationships between system components. Usually such systems can only be used to provide warnings when a failure has already caused a severe effect on the process performance.

Recently statistical analysis approaches became popular for the detection and diagnostics of faults of different industrial systems. The main advantage of these methods is that they do not need a detailed mathematical model of the system to perform the fault detection and diagnostics. Instead they are based on features of historical data obtained from these systems. Thus the same technique can be adapted to individual systems using their data. However, the biggest deficiency of statistical analysis approaches is that they need historical data of failure modes to be detected (Roverso, 2002). In some cases, such data is hard or even impossible to acquire.

Principal Component Analysis (PCA) was a common technique in the statistical analysis approaches category used for fault detection and diagnostics in the oil and gas industry. However, the results obtained from PCA usually need additional processing in order to perform fault detection. Moreover, individual PCA models had to be built for each failure mode from historical data, as mentioned previously. If multiple failures were considered, individual PCA models would also have to be built for each combination of failures. This could be a problem for a larger system in terms of the duration of the analysis.

The lack of detail in the results presented in the studies reported does not allow a clear judgement to be made as to their usefulness and performance. However, several deficiencies were identified:

- The model-based approaches tend to be developed for specific operating conditions, while detailed analysis is needed to adjust the model if the operating conditions change.
- The models are unable to detect multiple faults.

- Historical data of system faults is needed to build fault detection and diagnostics models. Such data might not always be available, specifically for faults that cause hazardous events.

Bayesian belief networks, as an alternative fault detection and diagnostics approach, which is introduced in the next section, can deal with all of the earlier mentioned deficiencies of the techniques presented.

3.2.2 Bayesian Belief Networks

A causal network that consists of a set of variables and a set of directed links which form a structure is called a directed graph. If there is a link from variable A to B, we say that B is a child of A, and A is a parent of B. Causal networks can be used to identify how a change of the certainty in one variable may affect the certainty of other variables. A Bayesian Belief Network (BBN) is a causal network, which has a quantitative representation of the causal links (Jensen and Nielsen, 2007). A BBN consists of the following:

- A set of variables and a set of directed edges between variables.
- Each variable has a finite set of mutually exclusive states.
- The variables together with the directed edges form an acyclic directed graph (abbreviated DAG). A directed graph is acyclic if there is no directed path $A_1 \rightarrow \cdots \rightarrow A_n$ so that $A_1 = A_n$.
- A conditional probability table $P(A|B_1, \dots, B_n)$ is attached to each variable A with parents $B_1, \dots, B_n$. If A has no parents, then the table reduces to the unconditional probability table $P(A)$.

A BBN lends itself to a graphical representation. An example BBN is given in Figure 3.3.

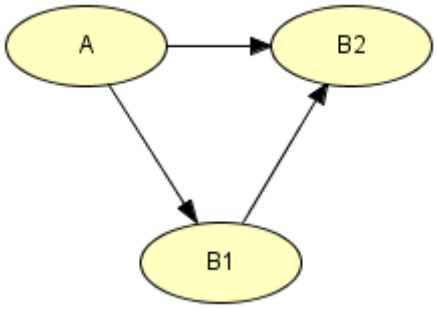


Figure 3.3. Example of a BBN

The BBN variables are represented as nodes $A, B1$ and $B2$, which are connected with the directed edges. The direction of an edge identifies the causal relationship enabling the parent and child nodes to be identified from the graphical representation of the BBN. For example, $B1$ and $B2$ are child nodes of A, while A is the parent node of $B1$ and $B2$. Moreover, $B1$ is the parent node of $B2$, thus $B2$ has two parent nodes. To define the BBN completely, an unconditional probability table $P(A)$ and conditional probability tables $P(B1|A)$ and $P(B2|B1, A)$ have to be specified. BBNs are usually built using expert knowledge.

3.2.2.1 Bayes theorem

The concept of the BBN technique is based on Bayes' rule, which provides a way to update beliefs about a certain variable B, given information about another variable A, using conditional probabilities. Variables A and B must have a finite set of mutually exclusive and exhaustive states, denoted as $(a_1, a_2, \dots, a_N)$ and $(b_1, b_2, \dots, b_M)$ respectively. The probability of variable A being in state a_i is denoted by $P(A = a_i)$, or, in short, $P(a_i)$. The conditional probability of variable B given variable A, $P(B|A)$, can be expressed as:

$$P(B|A) = \frac{P(A,B)}{P(A)}, \qquad \textbf{3.11}$$

where $P(A,B)$ is the joint probability of variables A and B, and is a $N \cdot M$ set of joint probabilities $P(a_i, b_j)$. Given a joint probability $P(A,B)$, the probability $P(A)$ can be found by marginalising variable B out of $P(A,B)$:

$$P(a_i) = \sum_{j=1}^{M} P(a_i, b_j), \qquad \textbf{3.12}$$

$$P(A) = \big(P(a_1), P(a_1), \dots, P(a_N)\big). \qquad \textbf{3.13}$$

The marginalising can be represented in short as:

$$P(A) = \sum_{B} P(A,B). \qquad \textbf{3.14}$$

Identically, the probability $P(B)$ is found by marginalising variable A out of $P(A,B)$. Having the notations defined, Bayes' rule is formed:

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)} = \frac{P(A,B)}{\sum_B P(A,B)}, \qquad \textbf{3.15}$$

where $P(B|A)$ is the posterior probability of B given A, $P(A|B)$ is the likelihood of A given B, $P(B)$ is the prior probability of B and $P(A)$ is the prior probability of A.

Similarly, the variable B might be conditioned on several variables, say an additional variable C, turning equation **3.15** into:

$$P(B|A,C) = \frac{P(A|B,C)P(B|C)}{P(A|C)} = \frac{P(A,B|C)}{\sum_B P(A,B|C)}, \qquad \textbf{3.16}$$

All of the probability updating in the BBN is performed using equation **3.16**.

3.2.2.2 Chain rule for Bayesian Belief Networks

Let $U = \{A_1, \dots, A_n\}$ be a universe of variables in the BBN. If we know the joint probability table $P(U) = P(A_1, \dots, A_n)$, then we can also calculate $P(A_i)$ as well as $P(A_i|e)$, where e is evidence about some of the variables in the BBN. However, $P(U)$ grows exponentially with the number of variables and the table becomes intractably large. Therefore, the general chain rule can be applied to the BBN. A unique joint probability distribution $P(U)$ can be given by the product of all conditional probability tables specified in the BBN:

$$P(U) = \prod_{i=1}^{n} P\big(A_i | \mathrm{pa}(A_i)\big), \qquad \textbf{3.17}$$

where $\mathrm{pa}(A_i)$ are the parents of A_i in the BBN, and $P(U)$ reflects the properties of the BBN.

The main purpose of the BBN is to update beliefs of occurrences of different node values in the BBN, after evidence to the variables has been introduced. Evidence e_i for variable A_i with N states is inserted as an N-dimensional vector of zeros and ones. After inserting evidence $e_1, \ldots, e_m$ for variables throughout a Bayesian network, the joint probability table can be updated using the chain rule (see equation **3.17**):

$$P(U, e) = \prod_{A \in U} P\big(A \big| \mathrm{pa}(A)\big) \prod_{i=1}^{m} e_i, \qquad \textbf{3.18}$$

and for $A \in U$ we have

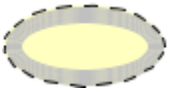
$$P(A|e) = \frac{\sum_{U \setminus \{A\}} P(U, e)}{P(e)} \qquad \textbf{3.19}$$

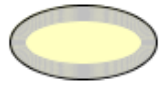
The initial probabilities of BBN variables are known as prior probabilities, while the updated probabilities after evidence was inserted into the BBN are known as posterior probabilities.

3.2.2.3 Graphical representation of BBN

Specialised software tools with a graphical user interface are often used to build a BBN graph. HUGIN software was used in this study to build and analyse BBNs. Different types of BBN nodes, which are used in this study are listed in Table 3.4.

Table 3.4. Types of BBN nodes used in the study

Node symbol	Node type	Description
	Discrete chance node	Discrete chance nodes are the basic and most-used nodes of the BBN. They can further be divided into different types: labelled, Boolean, numbered and interval. Labelled nodes are the ones that are used in this study. States of the labelled nodes are labels of a certain event or condition, e.g. labelled nodes are used to represent the working and failed states of components.
	Instance node	An instance node is used to create a modular BBN. If certain parts of the BBN are repetitive these can be constructed as instances and then used throughout the BBN. For example, an instance node can be created for the level control loop and if there are several level control loops with an identical structure, the instance node can be used to reduce the graphical complexity of the BBN. The BBN that has instance nodes is known as an Object Oriented Bayesian Network (OOBN).
	Input node	An input node is used to connect instance nodes to other nodes in the BBN. Input nodes must be chosen when a BBN, which is used as an instance node, is built. Input nodes

Node symbol	Node type	Description
		have to be of the same type and of the identical structure as the nodes of the BBN outside the instance node which are connected to the input nodes.
	Output node	An output node is used to provide an output (or multiple outputs) from the instance node.
	Information node	An information node is an additional type of a node derived to make a graphical representation of the BBN clearer. These nodes are used to provide evidence in the BBN. These nodes are coloured in pale green.
	Condition node	Nodes that represent states of every component (e.g. LT1 failure, LC1 failure) in the BBN are “Condition nodes”. These nodes are coloured in dark blue.

3.2.2.4 Bayesian Belief Networks for fault detection and diagnosis

Bayesian belief networks are widely used for modelling systems subject to a high level of uncertainty. The systems usually consist of a number of different components and the complexity and uncertainty lies in the interactions between those components. Some of the examples of the BBN method applied to such systems include aeroplane engines in (Sahin et al., 2007), a water tank system in (Lampis and Andrews, 2009b) and rotating flexible rotors in (Xu, 2012).

The BBN technique was combined with a particle swarm optimization (PSO) method for the fault detection and diagnostics of aeroplane engines in (Sahin et al., 2007). The proposed methodology consisted of three steps: pre-processing of input data, identifying the best structure of the BBN model with the PSO method and performing fault diagnostics with the BBN model.

Representative data was collected during a flight from various sensors measuring temperature, pressure and altitude. The available data was used in the PSO algorithm to generate a BBN with the best structure, based on the fault detection ability. A fault was considered as detected if the probability of the fault, given by the BBN, was bigger than a certain threshold.

A case study of an aeroplane engine was presented, where data from 15 oil-related sensors of the aeroplane engine were available. Various structures of BBNs were tested (fault nodes with different numbers of parents, nodes with different numbers of states). The best performing BBN obtained a 94.34% rate for correctly detecting faulty scenarios. However, 28.77% of the non-faulty scenarios were also detected as faulty. The high number of false alarms would result in unnecessary maintenance of the engine, if such an approach was used in practice. Nevertheless, the authors indicated that further tuning of parameters used in the PSO algorithm might lead to better results.

BBNs were used to detect and diagnose failures of a water tank system in (Lampis and Andrews, 2009b). This system somewhat resembles the system analysed in this thesis (both systems consisting of valves, controllers), however it is a relatively simple system, where only one liquid level has to be controlled and there are no

interactions between the flows from one subsection of the system to another, as observed in the three-phase separator.

The authors introduced a method to convert fault trees into BBNs. Four non-coherent fault trees were converted into BBNs. These networks were then connected to form a single network. The sensor readings from the water tank system were input as patterns in the BBN (for example constant flow, no flow, from constant to no flow and oscillating flow). In order to test the method proposed, a simulation model to simulate the operation of the water tank system was built using C++. 98.7% of the failures identified by the BBN corresponded to actual failure causes in the simulation model. The testing of the proposed approach showed the potential of the BBN technique to be used to detect and diagnose failures of similar systems.

A methodology based on BBN for the fault diagnosis of rotating machinery was proposed by Xu (Xu, 2012). Three layers were chosen to build the BBN: a machine running conditions layer (for example, uneven heating of rotors), a machine faults (for example, rotor crack) layer and a fault symptoms layer (for example, large magnitude of rotor vibrations). These three layers were interconnected in the BBN, which is given in Figure 3.4. Nodes belonging to machine running conditions, machine faults and fault symptoms layers are coloured in yellow, purple and green respectively.

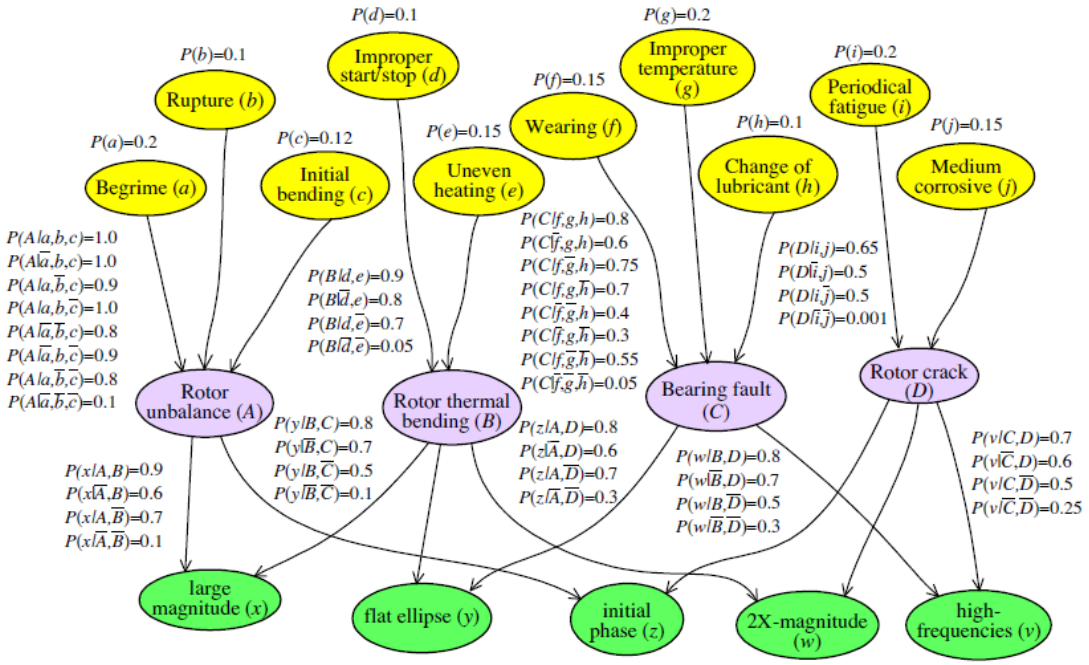


Figure 3.4. BBN for the fault diagnostics of rotating flexible rotors

The prior probabilities for the root nodes in the layer of machine running conditions were initially estimated by field experts and then refined by technicians according to the specific conditions of the machine. The conditional probabilities tables in the second and third layers were determined by experts.

Several scenarios with single and multiple faults were used to test the capabilities of the proposed BBN for the fault diagnostics. Faults were successfully diagnosed in 4 tested scenarios. However more extensive testing should be performed in order to validate the proposed methodology.

Overall, the BBN techniques used for fault detection and diagnostics have the following advantages:

- The modelled system is represented graphically, thus it is easier to visualise and understand the interactions between the components and subsystems of the whole system.
- Prior knowledge about the modelled system can be incorporated in the model based on historical statistical data or expert knowledge.
- The model can be developed and improved further without altering the structure of BBN, when new data about system components and their failure modes becomes available.
- The fault diagnostics can be performed iteratively. For example, an engineer is certain that some failure modes have not occurred. Evidence for these failure modes can be entered in the BBN to update the posterior probabilities of remaining failure modes.

Based on the success of previous studies and the mentioned advantages of the BBN modelling technique the methodology proposed in this study is also based on BBNs.

3.2.2.5 Summary

An extensive literature review has been carried out to identify the state of the art in fault detection and diagnostics used in the oil and gas industry. The majority of the approaches were based on mathematical models of the systems investigated (e.g. separators, stirring tanks) whose outputs were then compared to the performance data obtained from the systems. Residuals from these comparisons were then analysed in order to detect system failures. However, details of the results presented in the studies did not allow a meaningful evaluation of the performance of most of the model-based approaches to be performed.

The other big group of methods used for fault detection and diagnostics was statistical analysis methods, where the operational data of systems considered was used to perform a statistical analysis and classify the operation as faulty or fault-free. The main deficiency of such methods was the need to use operational data with faults present in the system for a training phase of the methods. Such data is usually not available in practice, especially for hazardous failure modes, since they occur rarely. Moreover, multiple failure modes were not considered in the studies investigated.

The Bayesian Belief Network technique was identified as a possible method to tackle the deficiencies of the previous approaches. An introduction to the technique was given and the chain rule for updating the posterior probabilities of the BBN was presented. Specific features, such as the colour coding used to identify different types of BBN nodes in the HUGIN software chosen to be used to build the BBNs, were described next.

Finally, several examples of fault detection and diagnostic techniques based on the BBN method were presented. Three different systems were considered, aeroplane engines, rotating flexible rotors and a water tank system. The performance of these techniques in the fault detection and diagnostics of the considered systems was discussed and their general advantages were identified. These showed a potential of the BBN method to be suitable for the fault detection and diagnostics of a three-phase separator.

The three-phase separator simulation model is presented in the next section.

3.3 Three-phase separator simulation model

When developing a fault detection and diagnostic technique for an industrial system, it is essential to test the technique in as close to real operating conditions as possible. This can be done in several ways: having a scaled version of the real operating system (Yang, 2006) or modelling the system and its operation using specialised software tools (Dias et al., 1993, Taylor and Omana, 2008). The first option is more desirable, since a scaled system can have operating conditions similar to those of the real system. However, this option is rarely used in practice, since building a test system can be costly. Moreover, testing using such a system usually takes more time, since all effects of failures have to be removed from the system before another failure can be inserted in the system. It might even be unsafe to induce specific failure modes, which might lead to the damage of the system.

The second option – system models – are favoured, since they can capture the main operating conditions of real systems and are cheap to implement. Furthermore, the data from the models is easily obtainable and even hazardous failure modes can be easily tested. The cost of developing models, the time taken to get the data and the ease of modelling different failures are the most important factors making the system models a preferred option for testing and validating fault detection and diagnostic techniques. The latter approach was also used in the research presented in this thesis.

Software with a graphical user interface for modelling the operation of a three-phase separator under both normal and affected by faults operating conditions was written in C++. A simplified model of a three-phase separator was considered. If necessary, the complexity of the simulation model can be increased by modifying model assumptions. The assumptions used for the three-phase separator modelling are as follows:

1. The separation process is assumed to be perfect. All of the incoming mixture is completely separated into three different phases, i.e. water, oil and gas, and the separation occurs instantly.
2. The layers of different phases are formed on top of each other and do not mix, just separate.
3. The volume of internal components (e.g. inlet diverter, weir) is not considered.
4. The maximum inflow is twice as high as an average inflow. Control valves are designed to allow the same maximum outflow rates as those of the maximum inflows, when the valves are fully open. When outlet valves are half opened, an amount corresponding to an average inflow is released from the vessel.
5. The precision of level transmitters is 1 cm, the precision of flow rate transmitters is 0.001 m^3 (1 litre) and the precision of pressure transmitters is 1 kPa.
6. Control valves can be opened with a 5% precision ranging from being fully closed (opened 0%) to fully opened (opened 100%).
7. A control command sent from the controller to a control valve is recalculated every 5 seconds, while levels of water, oil and gas are monitored every second.
8. There is no dead time between time points when the level of each phase is measured and sent from the transmitter to the controller; and then the control command is sent from the controller to the control valve.
9. Selected PI control parameters CO_b (controller bias value), K_C (controller gain) and T_i (integral time) are such that they do not cause a disruption to the operation of the three-phase separator when all of the control components are working.

10. Physical design parameters of the separator (e.g. weir height, separator radius etc.) and normal operating conditions (e.g. water-oil interface set point, oil level set point), given in Table 3.5 are assumed to be chosen appropriately.
11. Discrete 1 second time steps are used to model the operation of the three-phase separator.

Table 3.5. Parameters of the three-phase separator model

Parameter	**Value**
Separator radius	2 m
Gravity settling section length	7 m
Separated oil section length	3 m
Weir height	2 m
Water-oil interface set point	1 m
Oil level set point	1 m
Pressure set point	500 kPa
Maximum flow through valve CV1	0.15 m^3/s
Maximum flow through valve CV2	0.15 m^3/s
Maximum flow through valve CV3	0.15 m^3/s
Initial water level	1 m
Initial oil level	1 m
Initial pressure level	500 kPa

3.3.1 Simulating the operation of a three-phase separator

This section explains how the operation of the three-phase separator is simulated in the C++ model. Firstly, the structure of the C++ model to perform a single step of simulation of three-phase separator is presented in Figure 3.5. Note that the structure of the model involves only C++ functions and variables, which form a core for the simulation model. Additional functions and variables are used (mainly for inputting and outputting data) but are not given in the figure.

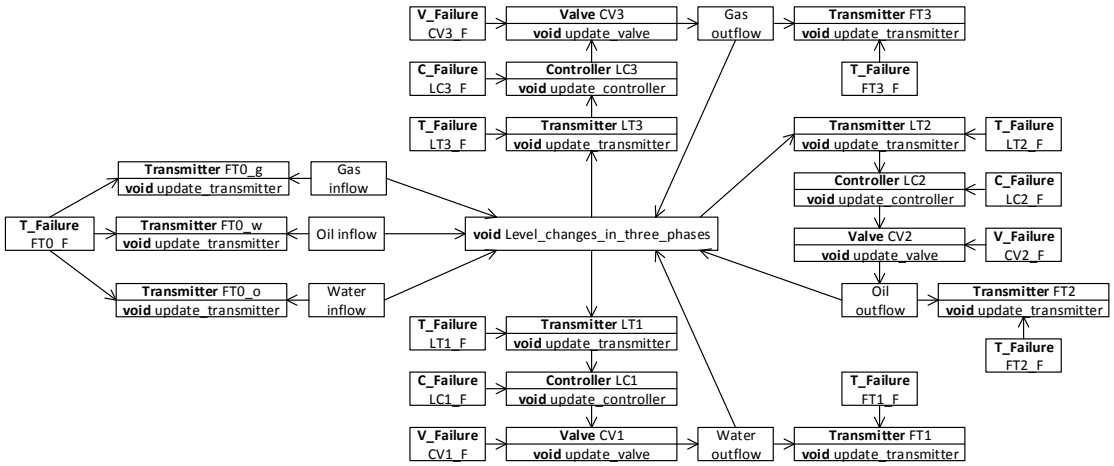


Figure 3.5. Structure of the three-phase separator simulation model in C++

The types of functions and variables are given in bold in Figure 3.5. The function that is responsible for the modelling of the water, oil and gas level changes in the separator is named “Level_changes_in_three_phases”. Once this function is executed, the execution of updating functions to update the information of components of three-phase separator is started.

Each component of the three-phase separator (see Table 3.2) is represented by a certain C++ structure of type “Transmitter”, “Controller” and “Valve”, given in Figure 3.6. Each structure has a function associated with it (“update_transmitter”, “update_controller” and “update_valve”) in order to update the information of variables in the structure.

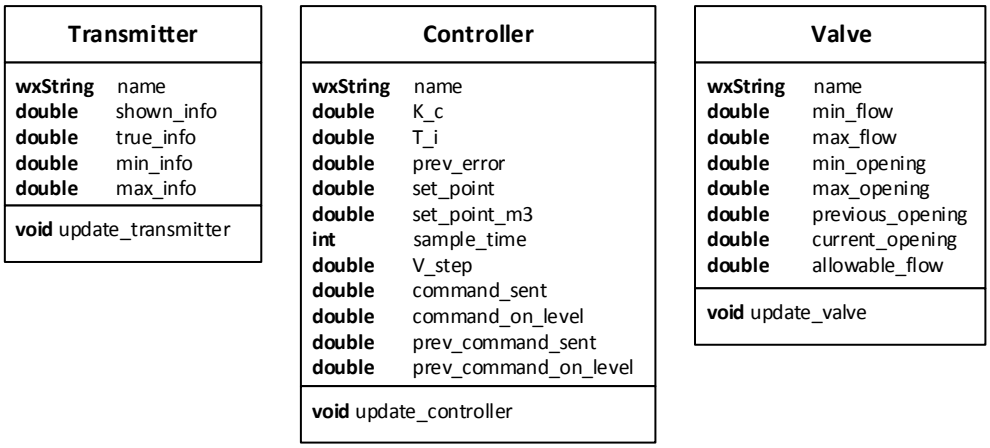


Figure 3.6. Types of structures in C++ to represent flow and level transmitters (Transmitter), PI controllers (Controller) and control valves (Valve)

The structures presented above are used in conjunction with additional structures that contain information about the failures of each component, as given in Figure 3.7. For example, a flow transmitter FT1 has an associated structure of type T_failure, where the wxString “name” in this structure is “FT1”, the Boolean variable “Fail” represents either faulty (“true”) or non-faulty (“false”) condition and integer variables “F_s_time” and “F_e_time” represent the start and end time of the considered failure mode.

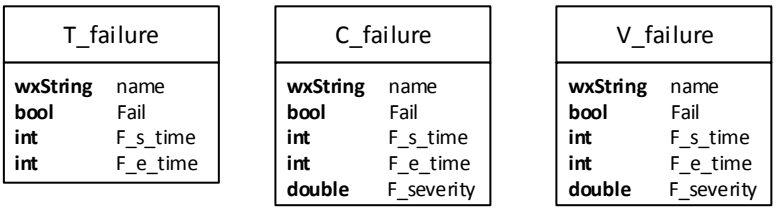


Figure 3.7. Types of structure in C++ to represent failure modes of transmitters (T_failure), controllers (C_failure) and valves (V_failure)

The above described structures form a core of the simulation model. The way the simulation is performed is presented next. The simulation can be summarised as given in the Figure 3.8.

Several input parameters have to be specified to initialise the separator model, for example the duration of simulation and the type of inflow into the separator (presented in section 3.3.2). Once the necessary parameters are initialised, the failure modes that will be simulated in the simulation run have to be specified. The start and end time points of the failure modes have to be set. However the rectification of component failures was not considered in this study, thus the end time of failure modes coincide with the end of the simulation run. The necessary inputs for the simulation model are set and the simulation can start.

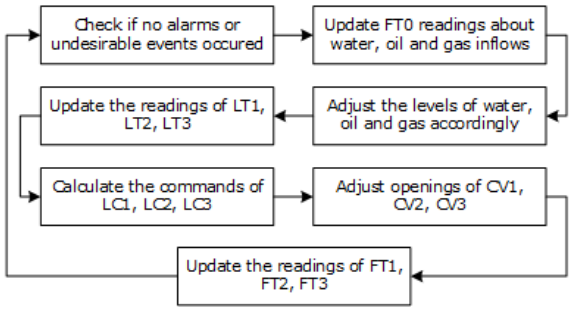


Figure 3.8. Simulating the operation of three-phase separator

At each time step of the simulation model the conditions for alarms (see section 3.3.7) or undesirable events (see section 3.1.2) are checked first. If none of the conditions are satisfied, the operation of the three-phase separator can be simulated further. The readings from transmitter FT0 about inflowing water, oil and gas are updated: the condition of transmitter FT0 is checked, and if the transmitter has not failed, the readings are set corresponding to true inflows. Otherwise, the readings are set corresponding to a certain failure mode. Next, the water, oil and gas levels are adjusted according to the inflowing and outflowing amounts of each phase. Once the levels are adjusted, the readings of level transmitters LT1, LT2 and LT3 are updated. If any of these transmitters failed, their readings are adjusted accordingly. Otherwise, the actual level, given the chosen precision of the level transmitter is set. The readings of level transmitters LT1, LT2 and LT3 are sent to the corresponding controllers LC1, LC2 and LC3.

The commands of PI controllers LC1, LC2 and LC3 are not continuously updated, since this helps to prolong the useful life of a valve. If the command of the PI controller has to be updated at the simulated time step, it is calculated based on the PI algorithm (see section 3.1.4) and depending on the state of the controller (working or in one of the failure modes) the respective command is sent to the corresponding valve.

The openings of valves CV1, CV2 and CV3 are adjusted in accordance to the commands sent by the corresponding controllers and the current states of the valves (working or in one of the failure modes). The readings of flow transmitters FT1, FT2 and FT3 are then updated, based on the flows through valves CV1, CV2 and CV3 and the condition of the flow transmitter (working or in one of the failure modes).

This procedure is repeated for the whole simulation duration or before an alarm is raised or an undesirable event occurs. Readings obtained from specific flow and level transmitters of the simulation model at each time step are then used to visualise the results of the simulation or to perform the fault detection and diagnostic process, which is introduced in section 3.4. More details on separate parts of the model are given in the next subsections, starting with the possible inflow options to be simulated.

3.3.2 Inflow options

Different inflow types are considered in the model of a three-phase separator in order to test whether the fault detection and diagnostics technique is able to cope with changing separator operating conditions. When different inflows of production fluids are considered, separated water, oil and gas levels can change due to variable inflow rates as well as component faults, thus making the fault detection and diagnostics more challenging. The types of liquid/vapour flows considered in this study are:

oscillating flow, increasing flow, decreasing flow and random flow. Details about these flows are given next.

3.3.2.1 Oscillating flow

An oscillating flow is modelled as a sine function. The minimum and maximum values of flow (LB and UB respectively) have to be specified together with a number of full sine waves, N_{waves}, throughout the simulation duration, N_{sim}. Then the flow can be simulated by the equation:

$$flow(t) = \frac{1}{2}(\sin(x) + 1)(UB - LB) + LB, \quad \textbf{3.20}$$

where $flow(t)$ is the simulated flow at time t and x is defined as follows:

$$x = \frac{2 \cdot N_{waves}}{N_{sim}} \cdot t \cdot \pi, \quad \textbf{3.21}$$

where N_{waves} is the number of full sine waves occurring during the simulation.

Using equations **3.20** and **3.21**, the simulated flow is ensured to have a selected number (N_{waves}) of full sine waves throughout the simulation duration (N_{sim}). The flow values range from LB to UB.

3.3.2.2 Increasing/decreasing flow

An increasing/decreasing flow is simulated in two different ways: by increasing/decreasing the flow linearly from the start to the end of the simulation and by increasing/decreasing the flow by certain step sizes. As in the case of the oscillating flow, the LB and UB values have to be specified. The linearly increasing/decreasing flow is simulated as follows:

$$flow(t) = flow(t-1) + sign \cdot \frac{(UB - LB)}{N_{sim} - 1}, t = 2,3, \dots, N_{sim}, \quad \textbf{3.22}$$

where $flow(t-1)$ is the simulated flow at time $t-1$, N_{sim} is the simulation duration and $sign = \begin{cases} -1, if\ decreasing\ flow \\ +1, if\ increasing\ flow \end{cases}$.

Note that, when $t = 1$, the starting conditions of the simulated flows are given by the following equation:

$$flow(1) = \begin{cases} LB, if\ increasing\ flow \\ UB, if\ decreasing\ flow \end{cases}. \quad \textbf{3.23}$$

The increasing/decreasing flow with discrete step sizes needs a number of steps N_{steps} to be defined. Then the simulation of flow can be described by the following equation:

$$flow(t) = \begin{cases} flow(t-1) + sign \cdot \frac{(UB - LB)}{N_{steps}}, if\ mod\left(t, \frac{N_{sim}}{N_{change}}\right) = 0 \\ flow(t-1), \quad otherwise \end{cases} \quad \textbf{3.24}$$

where $mod(x, y)$ is the modulus operation, which is the remainder of x divided by y.

The same equation as that in **3.23** holds for the step size increase.

3.3.2.3 Random flow

A random flow is simulated as a random variable with the Normal distribution. The necessary inputs are LB, UB and N_{change}. The mean μ for the Normal distribution is calculated as an average of LB and UB. The standard deviation σ of the Normal distribution can be calculated based on the three-sigma rule:

$$P(\mu - 3 \cdot \sigma \leq x \leq \mu + 3 \cdot \sigma) = 0.9973, \quad \textbf{3.25}$$

The three-sigma rule states that observation x, which is distributed with the Normal distribution $N(\mu, \sigma)$, lies within interval $[\mu - 3 \cdot \sigma; \mu + 3 \cdot \sigma]$ with 0.9973 probability. For example, if 10000 observations were generated using $N(\mu, \sigma)$, approximately 9973 of them would be within interval $[\mu - 3 \cdot \sigma; \mu + 3 \cdot \sigma]$. Thus the standard deviation can be calculated using this rule to generate random numbers, with the majority of them being in the $[LB, UB]$ interval, in the following way:

$$\mu - 3 \cdot \sigma = LB, \quad \textbf{3.26}$$

$$\mu + 3 \cdot \sigma = UB, \quad \textbf{3.27}$$

Given $\mu = \frac{LB+UB}{2}$, the standard deviation σ can be expressed from equations **3.26** and **3.27** as:

$$\sigma = \frac{UB - LB}{6}. \quad \textbf{3.28}$$

Then the flow is simulated as in the following equation:

$$flow(t) = \begin{cases} N(\mu, \sigma), if\ mod\left(t, \frac{N_{sim}}{N_{change}}\right) = 0 \\ flow(t-1), otherwise \end{cases}, t = 2,3, \dots, N_{sim} \quad \textbf{3.29}$$

Note that, the starting conditions of the simulated flow are given by the following equation:

$$flow(1) = N(\mu, \sigma). \quad \textbf{3.30}$$

Since 99.73% of the simulated values are within the wanted interval $[LB, UB]$ as given by the 3-sigma rule, the generated flows with equation **3.29** are bounded, to deal with the 0.27% of the values that are not within the wanted interval in the following way:

$$flow(t) = \min\big(\max\big(LB, flow(t)\big), UB\big). \quad \textbf{3.31}$$

3.3.3 Liquid flow through a valve

Valves are usually fitted as final control elements in the liquid level control loop. It is important to define how the flow of a liquid changes depending on the opening of the valve to be able to determine the expected flow with a given valve opening. Several valve characteristics influence an allowable liquid flow through it: a valve type (e.g. butterfly, globe, gate), a valve response time (time needed to change the openness of the valve by an indicated step size), valve inherent flow characteristic (e.g. linear, equal percentage, fast opening).

The most basic equation (Emerson Process Management, 2001) describing the flow of a liquid through a valve is:

$$Q_L = C_v \sqrt{\frac{P_1 - P_2}{G_f}}, \quad \textbf{3.32}$$

where Q_L is the volumetric flow rate through the valve in gallons per minute, C_v is the valve flow coefficient, which indicates the number of gallons per minute of water that will flow through the fully opened valve with 1 PSI of pressure drop, P_1 is the valve upstream pressure in PSI, P_2 is the valve downstream pressure in PSI and G_f is a liquid specific gravity (1 for water) having no units.

The flow through the valve is a non-linear function of a pressure drop across the valve, as can be seen from equation **3.32**. Thus, the flow through the valve not only depends on the opening of the valve, but also on the liquid level in the vessel and the working pressure of the vessel. However, this only holds if the liquid flows through the valve due to gravitational forces. In cases when the flow through the valve is regulated by a pump, the downstream pressure would be adjusted to obtain a required constant flow with a set valve opening angle. The latter valve operation scenario will be considered in this research. Thus the liquid flow through the valve will be a linear function of a valve opening and the maximum allowable flow:

$$Q_L = V_{open} Q_{Lmax}, \quad \textbf{3.33}$$

where V_{open} is the relative measure of valve opening (0-fully closed, 1-fully opened), Q_{Lmax} is the maximum allowable liquid flow through the valve.

3.3.4 Gas flow through a valve

Gas flow through a valve is different from that of liquids, since gas is compressible. When the gas experiences a pressure change its density also changes. Equation **3.32** for liquid flow through a valve has to be modified to describe the gas flow through a valve (Emerson Process Management, 2001):

$$Q_G = 963 C_v \sqrt{\frac{(P_1 - P_2)(P_1 + P_2)}{G_g T}}, \quad \textbf{3.34}$$

where Q_G is the gas flow through the valve in standard cubic feet per hour, C_v is the valve flow coefficient, also used in equation **3.32**, P_1 is the gas pressure upstream of the valve in PSIA, P_2 is the gas pressure downstream of the valve in PSIA, G_g is a specific gravity of gas (1 for air), T is the temperature of gas in degrees Rankine (°R).

However, this equation only holds when the gas stream velocity does not approach the speed of sound, i.e. when $P_2 \geq \frac{1}{2} P_1$ (Emerson Process Management, 2001).

The gas flow through the valve can be controlled to provide a constant flow of gas and a constant decrease of pressure in the vessel for a given constant valve opening. Instead of using a pump, a compressor or a blower can be used to regulate the gas flow. For the sake of simplicity, the gas flow will be regulated in this study, giving a linear relationship between valve opening and gas flow through the valve:

$$Q_G = V_{open} Q_{Gmax}, \quad \textbf{3.35}$$

where V_{open} is the relative measure of valve opening (0-fully closed, 1-fully opened), Q_{Gmax} is the maximum allowable gas flow through a valve.

3.3.5 Precision of transmitters

The precision of transmitters is implemented in the model in the following way:

$$T_{shown} = \frac{floor(k \cdot T_{true})}{k} \quad \textbf{3.36}$$

where T_{true} is the true information of the measured process variable, expressed in SI units, T_{shown} is the level indicated by a transmitter expressed in SI units, and k is the constant depending on the chosen precision, expressed as the ratio between one SI unit and the chosen precision in the SI unit, operation $floor$ is defined as:

$$floor(x) = \max_{m \in Z}(m \leq x), \quad \textbf{3.37}$$

where Z is the set of integers.

The precision of the level transmitter is 1 cm, giving $k = \frac{1\ m}{0.01\ m} = 100$. The precision of the flow rate transmitter is 0.001 m^3, giving $k = \frac{1\ \mathrm{m}^3}{0.001\ \mathrm{m}^3} = 1000$. The precision of the pressure transmitter is 1 kPa, giving $k = \frac{1\ Pa}{1000\ Pa} = 0.001$.

An example is given to show how the actual level is converted to a level indicated by a transmitter. Assume that a precise level is $T_{true} = 0.8152\ m$. Then, according to equation **3.36**, $T_{shown} = \frac{floor(k \cdot T_{true})}{k} = \frac{floor(81.52)}{100} = \frac{81}{100} = 0.81\ m$. Thus, when a precise level is $T_{true} = 0.8152\ m$, the transmitter will show it as $T_{shown} = 0.81\ m$.

One could argue that it is better to round the number instead of using the floor operation, since this would introduce smaller deviations from the actual values. However, the floor operation was chosen so that the level which has already been reached can be displayed. If the value was rounded, the transmitter would show level $T_{shown} = 0.82\ m$, before the actual level of 0.82 metres was reached.

3.3.6 Material movement in the separator

Flows of liquid and gas in the separator are modelled at discrete time steps of 1 second. Even though the movement of materials in the vessel happens simultaneously, it is modelled through discrete changes in their levels in the following order:

1. With the incoming fluid, the levels of liquids and the volume of gas in the gravity settling section are increased accordingly, i.e. the level of water gravity settling section, the level of oil in the gravity settling section and the volume of gas in the separator.
2. The water is the first to leave the separator through valve CV1 and the water level and the total liquid level in the gravity settling section are adjusted accordingly.
3. The flow over the weir is calculated based on the total liquid level in the gravity settling section.
4. The oil leaves the separator through valve CV2 and the oil level in the separated oil section is adjusted accordingly.
5. The volume of gas in the separator is calculated.
6. The gas leaves the separator through valve CV3 and the pressure is adjusted accordingly.

This process is repeated at each time step of the simulation.

3.3.7 Primary protection from undesirable events

Protection systems are used in order to avoid undesirable events, as mentioned in section 3.1.2. An alarm system is considered to be a protection system in this study. For each monitored variable (water-oil interface level, oil level and pressure) low and high level alarm thresholds are defined. The thresholds are set for the corresponding transmitters and when they are breached an alarm is raised. For the water-oil interface level the low level alarm threshold is denoted as WLLA (water low level alarm) and the high level alarm threshold is denoted as WHLA (water high level alarm). For the oil level the low level alarm threshold is denoted as OLLA and the high level alarm threshold as OHLA. Similarly, for the pressure, the low level alarm threshold is denoted as PLLA and the high level alarm threshold as PHLA. In this study the low level alarm thresholds WLLA and OLLA are chosen as equal to 20% of the weir height and the high level alarm thresholds WHLA and OHLA as 80% of the weir height. The PLLA is chosen as a deviation of -20% from a set point and the PHLA is chosen as a deviation of +20% from a set point. Values of alarm thresholds are given in Table 3.6.

Table 3.6. Alarm values for primary protection

Alarm	**Value**
Water-oil interface low level alarm (WLLA)	0.4 m
Water-oil interface high level alarm (WHLA)	1.6 m
Oil low level alarm (OLLA)	0.4 m
Oil high level alarm (OHLA)	1.6 m
Pressure low level alarm (PLLA)	400 kPa
Pressure high level alarm (PHLA)	600 kPa

It is considered that if an alarm is raised, appropriate preventative actions are performed and the undesirable event is avoided. If the alarm is not raised at an appropriate time the undesirable event occurs, given the operating conditions of the three-phase separator do not change in the meantime. For example PHLA is raised when level transmitter LT3 indicates a higher than 600 kPa value. Note that, if the transmitter has failed stuck indicating a lower level, an alarm will not be raised and an undesirable event of overpressure will occur. The simulation of the system operation is stopped if any of the alarms is raised or an undesirable event occurs.

3.3.8 Modelling failure modes

Component failure modes considered in the simulation model are associated with the automatic control and measuring of the liquid levels and flows of the three-phase separator and are given in Table 3.7.

Table 3.7. Failure modes of components of a three-phase separator

Component name	Failure mode	The way the failure mode is modelled
Flow transmitter (FT0, FT1, FT2, FT3)	1 – Failed stuck (FS)	The flow rate shown remains the same as at the time of failure irrespective of the actual flow rate.
Level transmitter (LT1, LT2, LT3)	1 – Failed stuck (FS)	The level shown remains the same as that at the time of failure and this level is sent to the controller each time the control command is updated, irrespective of the actual level.
PI controller (LC1, LC2, LC3)	1 – Failed low (FL)	The minimum output command is sent to the control valve, regardless of the actual information received from the corresponding transmitter.
	2 – Failed high (FH)	The maximum output command is sent to the control valve, regardless of the actual information received from the corresponding transmitter.
Control valve (CV1, CV2, CV3)	1 – Failed closed (FC)	The valve remains closed and prevents any flow, regardless of the actual command received from the controller
	2 – Failed opened (FO)	The valve remains fully opened and allows maximum flow, irrespective of the actual command received from the controller

Once a component has failed it is considered that it remains in that state until the end of the simulation and therefore it cannot fail again with a different failure mode. For example, if a controller has failed low it cannot fail high during the same simulation instance.

3.3.9 Graphical user interface

A graphical user interface was developed using a C++ library called wxWidgets (wxWidgets) to allow the user to interactively change physical parameters of the three-phase separator, choose the type of inflows to be modelled, insert failures in the simulation model and visualise the simulation results. A variety of physical parameters of separator (e.g. weir height, separation section length and etc.), different inflow options and the remaining input parameters (e.g. duration of simulation) can be set as shown in Figure 3.9.

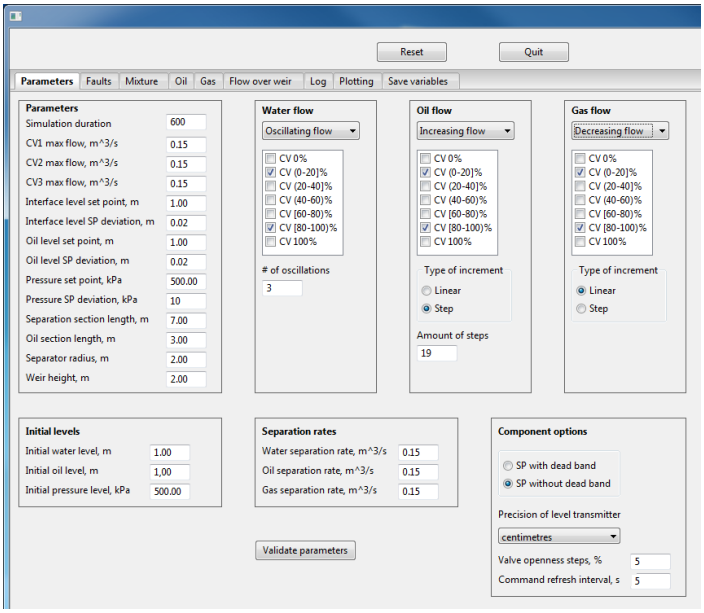


Figure 3.9. A graphical user interface to set the parameters of the three-phase separator model

Once these parameters are set, the next screen allows users to specify the faults to be modelled in the selected simulation run as shown in Figure 3.10. Here, a diagram of a three-phase separator is given so that the user could clearly see a position of the component in question in the separator and which section of the separator will be directly affected by a component fault.

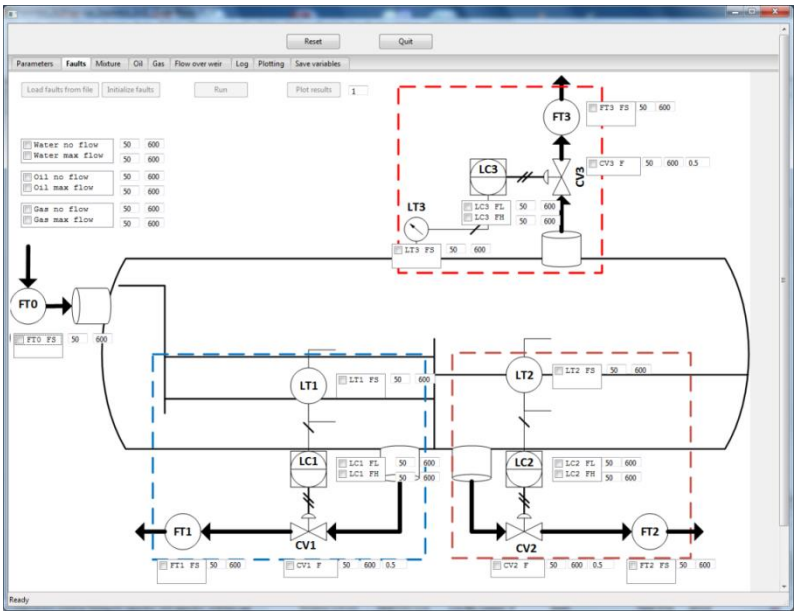


Figure 3.10. A graphical user interface to induce faults in the components of three-phase separator

After component faults have been selected (or imported from a data file) the simulation model can be run. Once the simulation has finished, changes in water, oil and gas levels in the separator can be plotted, as shown in Figure 3.11. The plots show the actual levels of liquids and the ones indicated by the level transmitters LT1, LT2 and LT3. The difference between actual levels and shown levels depend on the selected precision of level transmitters and their conditions, i.e. whether they are working or failed stuck.

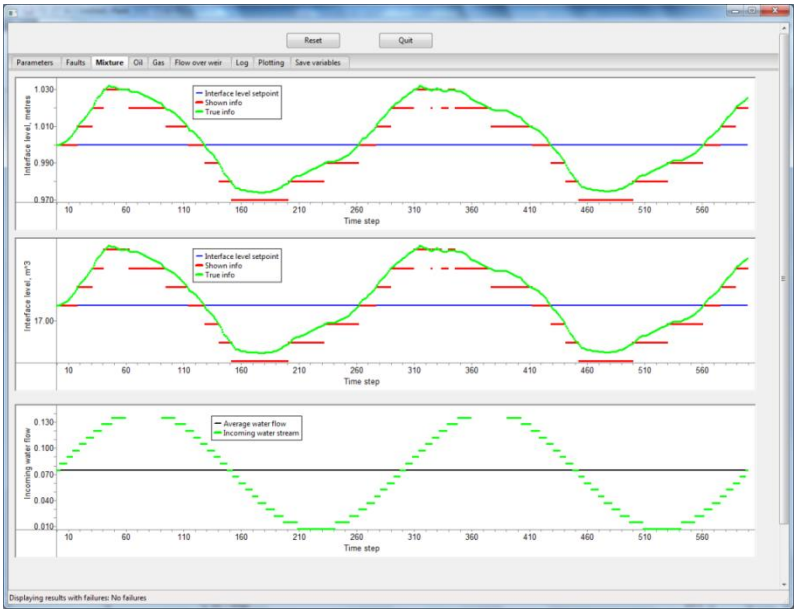


Figure 3.11. An example screen of the modelled water/oil interface level dynamics in the separation section given the oscillating water inflow

3.3.10 Summary

A common approach to test the fault detection and diagnostic technique is to use a simulation model of the operation of the considered system. Such an approach was chosen in this study and a simulation model of a three-phase separator was built. The main aim of the simulation model was to simulate the effects of component faults. The simulation model has a lot of flexibility built in. For example, several different types of oil, water and gas inflows can be modelled, different physical parameters of the separator can be chosen and different precisions of level transmitters can be set.

The core structure of the simulation model was presented first. This was followed by more detailed information about the specifics of the model, such as how different inflows were modelled, how the failure modes were inserted in the simulation model. Finally, a graphical user interface was presented, showing screenshots from the developed software and briefly introducing the sequence of the user input required to run the simulation model of the three-phase separator.

The information obtained from the simulation model was then used for the fault detection and diagnostics process, which is introduced in the next section.

3.4 Proposed methodology for the fault detection and diagnostics of three-phase separators

3.4.1 Overview of the proposed methodology

The proposed methodology for fault detection and diagnostics of a three-phase separator is based on the BBN technique. It models changes of water, oil and gas levels in the individual sections and the whole separator when the system is free from faults as well as when some faults exist. Since the three sections of the separator are highly dependable on one another, a failure in one section will usually have a noticeable impact on other sections. This can be detected if the measuring equipment has not failed in the remaining sections.

The idea of the methodology proposed for fault detection and diagnostics is to split the operation period of the three-phase separator into time intervals and then compare sensor readings recorded during several successive time intervals (called time slices in this study). The length for each time slice could be chosen depending on the operational conditions of the three-phase separator. Commonly used types of sensors are considered, such as flow transmitters and level transmitters. For the analysis purposes some pre-processing of the readings of transmitters is required in order to derive additional information about the processes in the separator (e.g. combination of flow rate transmitter readings) as well as capturing the dynamics of the separator (see section 3.4.2).

The BBN model was developed by replicating the way that water, oil and gas are propagated through each section before they leave the separator. The nodes in the BBN model represent conditions of three-phase separator components, sensor readings and changes in levels of oil, gas and water. Two types of nodes are defined: condition and information nodes. Condition nodes are used to represent states of components. Information nodes are used to input sensor readings into the model. The arcs are used to model the causal relationships between the nodes. For example, arcs are used to join the respective nodes to model the effect of the component failure on sensor readings.

The BBN for the fault detection and diagnostics was developed in a modular way. First of all, generic rules were formed and a BBN called *"FT_ID given LT_ID"* was built to model a control loop in a three-phase separator (see section 3.4.3.1). Then generic rules were formed for constructing a BBN to model the liquid or vapour changes in a generic section (see section 3.4.3.2) which also includes the previously built BBN *"FT_ID given LT_ID"*. Due to distinctive features of each section, section specific rules were also developed in addition to the generic rules and individual BBNs were built for each section. The BBNs *"Separation section"*, *"Separated oil section"* and *"Gas section"* can be found in sections 3.4.3.3, 3.4.3.4 and 3.4.3.5, respectively.

Once the BBNs for the individual sections were built, a BBN to model interactions between these sections was developed, combining individual section BBNs and was called a *"Single time slice"* BBN (see section 3.4.3.6). It models changes of water, oil and gas levels in all sections during an observed time slice. Multiple instances of the *"Single time slice"* BBNs were then combined (adding notation to each "t-3=>t-2", "t-2=>t-1" and "t-1=>t") to obtain a *"Triple time slice"* BBN, which was used to perform the fault detection and diagnostics stage. The structure of the final BBN model is given in Figure 3.12. When BBNs are built in this way (using BBNs as a part of another BBN), they are referred to as Object Oriented Bayesian Networks (OOBN).

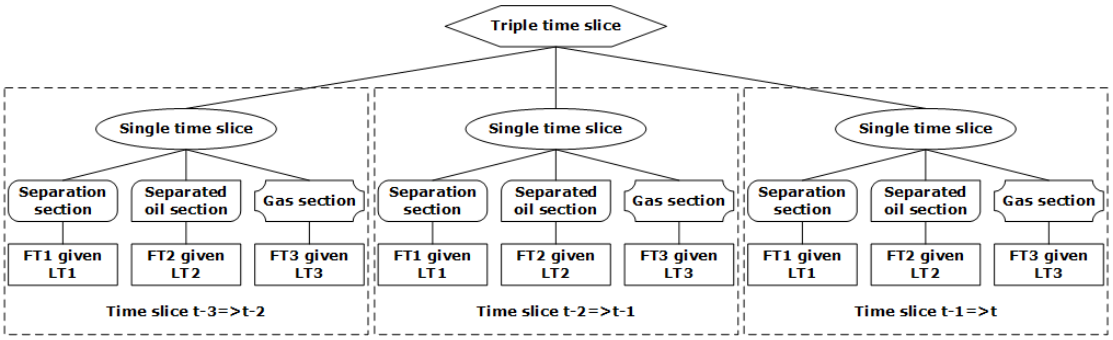


Figure 3.12. Structure of the whole system OOBN model *"Triple time slice"* used for fault detection and diagnostics

Note that specific shapes representing individual parts of the system OOBN in Figure 3.12 are used to denote BBNs with identical structure (for example, FT1 given LT1 and FT2 given LT1).

When simulating the operation of the system, the posterior probabilities of the states of condition nodes (nodes representing conditions of components, see section 3.2.2.3) in the last slice of the *"Triple time slice"* OOBN (namely *"Single time slice t-1=>t"*) are being monitored during the simulation time. A certain component failure mode is considered to be detected if the posterior probability representing that mode exceeds a certain threshold. Thus the fault detection and diagnostics stage is performed simultaneously. Note that, whenever the fault detection and diagnostics model will be mentioned in the further sections, it will represent the OOBN *"Triple time slice"*.

The developed methodology is presented in the next sections in more detail.

3.4.2 Fault detection and diagnostics model input data

As mentioned previously, there are components in the three-phase separator, which provide a way to monitor the operation of the separator. Flow rate transmitters and level transmitters (which are primarily used as part of process control equipment) are the components which can provide the information needed for fault detection and diagnostics purposes. Their provided readings are used as input data for the OOBN model. In the proposed methodology this data is obtained from the simulation model. The majority of information available from transmitters in the separator simulation model is pre-processed before it can be inputted into the OOBN. Moreover, not only readings from single transmitters but also several combinations of transmitter readings are considered in order to derive additional information to be used in the OOBN. The transmitter data used in the OOBN and its pre-processing is discussed in more detail next.

3.4.2.1 Combining the information from the transmitters

Combining the information from multiple transmitters installed on the separator allows supplementary data about the processes in the separator to be obtained without a need to install additional equipment. For example, in the separation section the combination of transmitter FT0 and FT1 readings can be used to determine the water level change rate, since FT0 transmitter has information about the inflowing water and FT1 transmitter has information about the outflowing water. Moreover, the water level change rate can also be determined from the readings of transmitter LT1. Thus,

by comparing the data of combined transmitters FT0 and FT1 and single transmitter LT1 the actual water level change rate can be evaluated and failure of one of the transmitters can be detected. Combinations of the transmitter readings (addition or subtraction of values) considered in this study are listed in Table 3.8.

Table 3.8. Combinations of transmitter readings

Combination	**Purpose**
"FT0 water – FT1"	Determines the water level change rate in the separation section and aids the fault detection of transmitters FT0, FT1 and LT1
"FT0 water – FT1 + FT0 oil"	Determines the total liquid level change rate in the separation section and aids the fault detection of transmitters FT0, FT1 and LT1
"FT0 gas – FT3"	Determines the gas level change rate in the separator and aids the fault detection of transmitters FT0, FT3 and LT3
"FT0 water - FT1 + FT0 oil - FT2"	Determines the volume change for gas in the separator and aids the fault detection of transmitters FT0, FT1, FT2, LT1 and LT2
"LT1 total liquid level change + LT2 level change"	Determines the volume change for gas in the separator and aids the fault detection of transmitters FT0, FT1, FT2, LT1 and LT2

The use of the listed combinations of transmitter readings improves the accuracy of fault detection. For example, if the "LT1 total liquid level change + LT2 level change" and the "FT0 water – FT1 + FT0 oil – FT2" readings differ, it indicates that one of the components FT0, FT1, FT2, LT1 or LT2 might have failed.

3.4.2.2 Pre-processing the readings from the transmitters

As mentioned previously, a pre-processing of readings from transmitters is performed before this information is used as input data of the OOBN. The available information is summarised in Table 3.9, giving the name of the information and the type of the pre-processing that is performed on it. Note that readings from the same transmitters may be pre-processed in several ways. The different pre-processing options, PP01-PP06, performed in this study are presented in the next subsections.

Table 3.9. Sensor readings as inputs to the OOBN model

Information name	**Pre-processing**
FT0 water inflow info	PPO1, PPO4 (CV1), PPO6
FT0 oil inflow info	PPO1, PPO4 (CV2), PPO6
FT0 gas inflow info	PPO1, PPO4 (CV3), PPO6
LT1 info	PPO2, PPO1, PPO6
LT2 info	PPO2, PPO1, PPO6
LT3 info	PPO2, PPO1, PPO6
FT1 info	PPO1, PPO4 (CV1), PPO6
FT2 info	PPO1, PPO4 (CV2), PPO6
FT3 info	PPO1, PPO4 (CV3), PPO6
FT0 info varied	PPO5, PPO6
FT1 info varied	PPO5, PPO6
FT2 info varied	PPO5, PPO6
FT3 info varied	PPO5, PPO6
LT1 total liquid level	PPO1, PPO6

Information name	Pre-processing
LT1 water level change rate	PPO3, PPO4 (CV1), PPO6
LT1 total liquid level change rate	PPO3, PPO4 (CV1), PPO6
LT2 oil level change rate	PPO3, PPO4 (CV2), PPO6
LT3 pressure level change rate	PPO3, PPO4 (CV3), PPO6
FT0 water – FT1	PPO1, PPO4 (CV1), PPO6
FT0 water – FT1 + FT0 oil	PPO1, PPO4 (CV1), PPO6
FT0 gas – FT3	PPO1, PPO4 (CV3), PPO6
FT0 water – FT1 + FT0 oil – FT2	PPO1, PPO4 (CV1+CV2), PPO6
LT1 total liquid level change + LT2 level change	PPO3, PPO4 (CV1+CV2), PPO6

3.4.2.2.1 Averaging at selected sampling intervals (PPO1)

Readings obtained from flow and level transmitters has to be averaged at certain sampling times (denoted as pre-processing option 1, PPO1) in order to present the information about a flow/level during a certain time period in the format compatible with the OOBN requirements. This is performed by dividing the information from transmitters into sampling intervals. For example, if the sampling time is every 10 seconds, the information from a transmitter is then extracted every 10 seconds, giving 10 readings (if the transmitter records one reading per second) which are then averaged. Thus the averaging at selected sampling intervals can be summarised as follows:

1. Extract the relevant information from a selected transmitter for the chosen sampling time.
2. Calculate the average of the information based on the formula:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i, \qquad \textbf{3.38}$$

where $\bar{x}$ is the averaged reading from the selected transmitter, x_i is the reading from the selected transmitter at the ith point of the sampling interval, n is the number of time points in the sampling interval.

3.4.2.2.2 Transforming into a corresponding PI command (PPO2)

The information from level transmitters measuring water, oil or pressure levels is transformed into a corresponding PI command by mimicking the PI controller algorithm (denoted as pre-processing option 2, PPO2). This is done in order to determine the exact command that would be sent to a control valve, given the readings of a level transmitter if the controller is functioning correctly. For example, if level transmitter LT1 shows a water level of 1 meter, the PI controller algorithm, as defined in **3.1**, is used to determine what command (CO) would be indicated by the controller with the given LT1 level. So the information provided by level transmitters is transformed into a PI controller command, specifying how much the valve should be opened for a given level of the substance measured.

3.4.2.2.3 Calculating the difference at selected sampling intervals (PPO3)

The difference of the level transmitter readings is calculated in order to provide the OOBN model with information on how the levels of substances measured change over time. As in the case of averaging transmitter readings, the information from a

transmitter is divided into sampling intervals first. The difference between the end and start points of the sampling interval is then calculated (denoted as pre-processing option 3, PPO3) based on the formula:

$$x_{diff} = \frac{x_n - x_1}{n}, \quad \textbf{3.39}$$

where x_{diff} is the averaged difference of readings from the transmitter, x_n is a reading from the transmitter at the end point of the sampling interval, x_1 is a reading from the transmitter at the start point of the sampling interval.

3.4.2.2.4 Norming to the maximum (PPO4)

The norming of the transmitter readings in this study (denoted as pre-processing option 4, PPO4 (name of the corresponding valve)) is usually performed after the averaging or difference calculation of the transmitter readings has been performed. For example, norming of flow transmitter readings is performed to transform the readings into a relative opening of the corresponding valve. The norming of the information is performed based on the formula:

$$x_i^{norm} = \frac{x_i}{x_{max}} \cdot 100\%, \quad \textbf{3.40}$$

where x_i is the actual information, x_{max} is the specified maximum, x_i^{norm} is the normed to the maximum reading.

Since each transmitter is associated with a respective valve, the maximum value for norming is defined as the maximum possible flow through that valve. For example, readings from FT0 on incoming water are normed according to the maximum flow allowed by valve CV1.

3.4.2.2.5 Calculating variance at selected sampling intervals (PPO5)

A variance of flow rate transmitter readings (denoted as pre-processing option 5, PPO5) is calculated at selected sampling intervals. It is done in order to input the information about the changes of flow during a selected time interval into the OOBN and aid the fault detection and diagnostics of flow transmitters. The information from the transmitters is divided into sampling intervals and then the variance is calculated based on the formula:

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n} x_i - \bar{x}, \quad \textbf{3.41}$$

where σ^2 is variance, n is the number of readings in the sampling interval, x_i is the ith reading, $\bar{x}$ is the average of readings in the sampling interval as defined by equation **3.38**.

3.4.2.2.6 Discretising into selected states (PPO6)

All of the readings from transmitters are discretised (denoted as pre-processing option PPO6), since only discrete nodes are used in our OOBN. Discrete intervals are defined separately for individual transmitters, based on their association with respective valves, and are then used as states of respective nodes in the OOBN. Simple if-then rules are used to assign the pre-processed information from the readings of the transmitters to these discrete states. The specific discrete intervals

and corresponding states of the nodes will be presented when the OOBN model and individual BBN models are described in later sections.

3.4.3 Fault detection and diagnostic OOBN model for the three-phase separator

As mentioned at the beginning of section 3.4.1, the OOBN model for fault detection and diagnostics was built in a modular way. In this section the BBN models that were developed as modules of the OOBN model are discussed in more detail and the procedure of fault detection and diagnostics of a three-phase separator with the OOBN is given.

First, the model for a control loop is presented, giving a set of rules to build the BBN called *"FT given LT"*. Next, some generic rules to build a BBN model for a section are given. Using these rules, the instances of previously developed BBN *"FT given LT"* and some section specific rules, BBN models for each individual section, named as *"Separation section"*, *"Separated oil section"* and *"Gas section"*, respectively are developed. The methodology how to combine BBNs for three sections into a larger BBN called *"Single time slice"* to model the interactions between those sections is provided next. Finally, it is explained how three instances of *"Single time slice"* BBNs are joined to obtain a *"Three time slice"* OOBN, which is used for fault detection and diagnostics of three-phase separator.

3.4.3.1 BBN model for a control loop

Firstly, a BBN is built to model the flow through a valve. The order in which information is circulated between components is exploited to decide which nodes are parent nodes and which ones are child nodes in the constructed control loop BBN. The information is circulated in the following order: level transmitter LT -> controller LC -> control valve CV -> flow transmitter FT. For example, a control signal sent from the controller LC to the control valve CV depends on the information received by the controller from the transmitter and the condition of the controller. Thus a node representing the information the controller receives from the transmitter and the node representing the condition of the controller are made parent nodes of a node representing the control signal sent from the controller LC to the control valve CV.

A generic sequence of steps can be determined to model the flow through a valve with a BBN:

Step 1. Create a node to represent all the possible levels of the process variable (for example, water level, separated oil level) obtained from transmitter LT ("LT info" node in Figure 3.13). Assign a prior probability table for the node.

Step 2. Create a node with states representing all the possible failure modes of transmitter LT as well as a state for the working mode ("LT failure" node in Figure 3.13). Assign the prior probabilities for the failure modes. The prior probability of working mode is obtained by subtracting the probabilities of all failure modes from 1.

Step 3. Create a node with states representing all the possible levels of a process variable transformed into the PI output (as explained in section 3.4.2.2.2) which are sent to controller LC ("LT sends info to LC" node in Figure 3.13).

Step 4. Make nodes “LT info” and “LT failure” the parents of node “LT sends command to LC”. Specify the conditional probability table for node “LT sends info to LC”, by considering how a certain failure mode from “LT failure” influences the information (obtained from “LT info”) sent to the controller.

Step 5. Create a node with states representing all the possible failure modes of controller LC together with a state for a working mode (“LC failure” node in Figure 3.13). Assign the prior probabilities for the states.

Step 6. Create a node with states representing all the commands that can be sent to valve CV (“LC sends command to CV” node in Figure 3.13).

Step 7. Make nodes “LT sends command to LC” and “LC failure” the parents of node “LC sends command to CV”. Specify the conditional probability table for node “LC sends command to CV”, by considering, how a certain failure mode from “LC failure” influences the command (obtained from “LT sends info to LC”), sent to the valve.

Step 8. Create a node with states representing all the possible failure modes of valve CV together with a state for working mode (“CV failure” node in Figure 3.13). Assign the prior probabilities for the states.

Step 9. Create a node with states representing all the possible openings of valve CV (“CV opening” node in Figure 3.13).

Step 10. Make nodes “LC sends command to CV” and “CV failure” the parents of node “CV opening”. Specify the conditional probability table for node “CV opening”, by considering, how a certain valve failure mode from “CV failure” influences the opening of the valve, which should be given by “LC sends command to CV”.

Step 11. Create a node with states representing all the possible failure modes of flow transmitter FT together with a state for working mode (“FT failure” node in Figure 3.13). Assign the prior probabilities for the states.

Step 12. Create a node with states representing all the possible flows through valve CV (“FT info” node in Figure 3.13).

Step 13. Make nodes “CV opening” and “FT failure” the parents of node “FT info”. Specify the conditional probability table for node “FT info”, by considering, how a certain flow transmitter failure mode from “FT failure” influences the information which should be shown by the flow transmitter, given the state of node “CV opening”.

The BBN, developed following these steps, was used as an instance node (see description in section 3.2.2.3) in the BBNs developed for specific sections of three-phase separators. The graphical representation of the developed BNN is given in Figure 3.13. Note that different shades of nodes represent different types of nodes, as explained in section 3.2.2.3. The details of the BBN nodes can be found in Table 10.1 (see Appendix E).

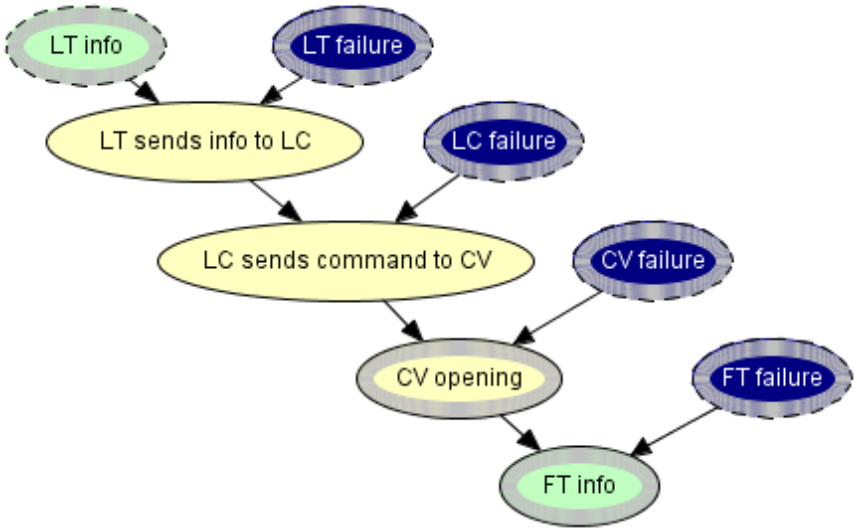


Figure 3.13. BBN "*FT_ID given level LT_ID"* for a control loop

3.4.3.2 Generic rules to build a BBN model for a section

The concept of building a BBN for an individual section of the separator is based on the representation of the propagation of liquid or vapour flows in that section. First an inflow of the material to the selected section and inflow measurements (if the inflow is measured) are modelled. Nodes representing sensor readings of flows/levels are made child nodes of nodes representing the true flows/levels and conditions of the components, which measure flows/levels. Similarly, the outflow of liquid or vapour from the selected section is modelled. Each section has an instance node "*FT given level LT"*, i.e. a BBN that represents a control loop. The inflows and outflows of the liquids or vapour are compared by looking at how the level of liquid or vapour changes given inflow to the section and outflow from the section. Nodes that model the liquid/vapour level changes and the ones that represent conditions of components which measure the liquid/vapour level changes are made parent nodes of the nodes that represent sensor readings about the liquid/vapour level changes. The generic rules to build a BBN for a section are given next.

Rule 1. Create a node labelled *"Liquid/vapour inflow"* to represent a flow of each type of liquid/vapour. Specify all possible discrete levels of the flow as states of this node, expressed as a percentage of maximum outflow of a corresponding valve (used to release the liquid/vapour from this section or the vessel itself). Specify prior probabilities for the states of the node: if information about the inflowing liquid/vapour is known, the prior probability for the corresponding state(s), representing that flow can be increased, otherwise prior probabilities of each state are made equal.

Rule 2. Create a node labelled *"FT_ID liquid/vapour inflow failure"* to represent a condition of each transmitter, measuring the inflowing liquid/vapour (if a transmitter is present). The working condition is set as the first state of the node, while the remaining states represent failure mode(s). Prior probabilities for each state have to be specified.

Rule 3. Create a node labelled *"FT_ID liquid/vapour inflow info"* to represent readings from a transmitter (if a transmitter is present) for every type of liquid or vapour flowing to the section. States of this node are identical to those of *"Liquid/vapour inflow"*. Make the nodes *Liquid/vapour inflow"* and *"FT_ID*

liquid/vapour inflow failure" parents of the node *"FT_ID liquid/vapour inflow info".* A conditional probability table of the node is then used to model the effects on the readings of the transmitter based on the condition of the transmitter and the rate of the inflowing liquid/vapour.

Rule 4. Create an instance node labelled *"FT_ID given level LT_ID"* (see the BBN model in Figure 3.13 in section 3.4.3.1) to model a control loop for any liquid/vapour level that is controlled.

Rule 4.a. Create a node labelled *"LT_ID info"* to represent readings from level transmitter converted to a PI command and make it a parent of input node *"LT info".* Set equal prior probabilities for each state.

Rule 4.b. Create a node labelled *"LT_ID failure"* to represent the condition of the level transmitter and make it a parent of input node *"LT failure"*. Set prior probabilities for each state.

Rule 4.c. Create a node labelled *"LC_ID failure"* to represent the condition of the level controller in the control loop and make it a parent of input node *"LC failure"*. Set prior probabilities for each state.

Rule 4.d. Create a node labelled *"CV_ID failure"* to represent the condition of the control valve in the control loop and make it a parent of input node *"CV failure"*. Set prior probabilities for each state.

Rule 4.e. Create a node labelled *"FT_ID failure"* to represent the condition of the flow transmitter indicating the flow through the control valve in the control loop and make it a parent of input node *"FT failure"*. Set prior probabilities for each state.

Rule 4.f. Create a node labelled *"FT_ID info"* to represent readings of the flow transmitter measuring the flow through the control valve in the control loop and make the output node labelled *"FT info"* a parent of the created node. The conditional probability table must have zero entries everywhere, except the diagonal, which is formed of ones.

Rule 4.g. Create a node labelled *"Liquid/vapour outflow"* to represent the flow through the control valve in the control loop and make the output node labelled *"CV opening"* a parent of the created node. The conditional probability table must have zero entries everywhere, except the diagonal, which is formed of ones.

Note that, the parent and child nodes in the Rule 4 have to have identical states.

Rule 5. Create a node(s) labelled *"Liquid/vapour change rate"* to model changes of every liquid/vapour in the section, using given liquid/vapour inflow and outflow. This node is a child node of earlier created nodes *"Liquid/vapour inflow"* and *"Liquid/vapour outflow"* (for each liquid/vapour). The conditional probability table of *"Liquid/vapour change rate"* specifies what the liquid/vapour change rate will be for the given liquid/vapour inflow and outflow.

Rule 6. Create a node labelled *"FT_ID liquid/vapour – FT_ID"* to represent the combined information from flow transmitters indicating liquid/vapour inflow and outflow. This node is made a child node of earlier created nodes *"Liquid/vapour inflow"*, *"Liquid/vapour outflow"*, *"FT liquid/vapour inflow failure"*

and *"FT_ID failure"*. The conditional probability table of *"FT_ID liquid/vapour – FT_ID"* specifies what combined readings of the transmitters will be for their given conditions.

Rule 7. Create a node labelled *"LT_ID liquid/vapour level change rate"* to represent the information of liquid/vapour change rate given by level transmitter LT_ID. This node is made a child node of earlier created nodes *"Liquid/vapour change rate"* and *"LT_ID failure"*.

Note that labels of nodes, which have symbols *"_ID"*, are used to represent components that are available in the section and the labels of nodes, which have symbols *"Liquid/vapour"* correspond to the type of liquid or vapour considered. For example, the labels of nodes in Rule 4.a, Rule 4.b, Rule 4.c, Rule 4.d and Rule 4.e in the separated oil section will be *"LT2 info"*, *"LT2 failure"*, *"LC2 failure"*, *"CV2 failure"* and *"FT2 failure"* respectively and the label of the node in the Rule 4.g will be *"Oil outflow"*.

These rules are the basis for creating a BBN for each section of the three-phase separator. Since all three sections are different, some alterations are required to the generic rules to suit specific needs of each section. For example, the separated oil section does not have a flow transmitter to measure the incoming oil from the separation section, thus Rule 2, Rule 3 and Rule 6 does not apply to this section. The BBNs for each section are presented next.

3.4.3.3 BBN model for the separation section

The BBN for the separation section was developed using the generic rules, described in section 3.4.3.2. Additional nodes *(*namely *"Liquid difference"*, *"FT0 water – FT1 + FT0 oil"*, *"Total liquid level"*, *"LT1 total liquid level"*, *"Flow over weir"*, *"Total liquid level change rate"*, *"LT1 total liquid level change rate"*) had to be also included in the BBN, since this section has an overflow weir, which controls the maximum level of the liquid in the section and the amount of oil flowing into the separated oil section. The additional nodes were created and integrated into the BBN in the following manner:

Step 1. A node labelled *"FT0 water – FT1 + FT0 oil"* was created in a similar way as described in Rule 6 in section 3.4.3.2. However, this time a combination of readings of water and oil inflows together with water outflow was used. Such a combination provides information about the liquid changes (difference between inflowing water and oil and outflowing water) in the separation section. The nodes labelled *"Water inflow"*, *"Water outflow"*, *"Oil inflow"*, *"FT0 failure"* and *"FT1 failure"* are made parents of the created node. The conditional probability table of the created node shows what the readings of the combined transmitters are when actual liquid change and conditions of flow transmitters FT0 and FT1 are given.

Step 2. A node labelled *"Total liquid level"* was created in order to aid the modelling of flow over the weir. States of this node represent discrete levels of the total liquid level, which determines the amount of liquid flowing over the weir. For example, if the total liquid level is below the weir by an amount which is greater than the maximum liquid increase in the separation section, then there would be no flow of liquid to the separated oil section. The prior probabilities for each state of the node should be set. Since during normal operation of the separator it is expected that the total liquid level is equal with the weir, this should be reflected in the prior probabilities.

Step 3. A node labelled *"LT1 total liquid level"* was created to input total liquid level readings from transmitter LT1. This node has the same states as the node labelled *"Total liquid level"*. Nodes labelled *"Total liquid level"* and *"LT1 failure"* are made parents of this created node. The conditional probability table of this node shows what readings from transmitter LT1 are expected for the given actual total liquid level and the condition of transmitter LT1. Thus, it can be used to model the effects of LT1 failure modes on the readings of the total liquid level.

Step 4. A node labelled *"Flow over weir"* was created to model the flow of oil into the separated oil section based on the total liquid level and the liquid changes in the section. Previously created nodes labelled *"Total liquid level"*, *"Water inflow"*, "Water outflow" and "Oil inflow" are made parents of this node. The conditional probability table of this node shows what flow is expected over the weir given the total liquid level and the liquid changes in the section.

Step 5. A node labelled *"Total liquid level change rate"* was created to model a change in the total liquid level in the separation section. It is expected that the total liquid level would not change under the normal operation of the separator and stay equal with the weir. Nodes labelled *"Total liquid level" "Water inflow"*, "Water outflow" and "Oil inflow" are made parents of this node. The conditional probability table of this node shows how the total liquid level changes given the total liquid level and the liquid difference in the section.

Step 6. A node labelled *"LT1 total liquid level change rate"* was created to input readings of a total liquid level change rate and to model the effects of LT1 failure on these readings. Nodes labelled *"Total liquid level change rate"* and *"LT1 failure"* are made parents of this node. The conditional probability table of this node shows how the condition of LT1 transmitter affects the readings of the total liquid level change rate.

The BBN that was developed for the separation section can be seen in Figure 3.14. The details of the BBN nodes can be found in Table 10.2 (see Appendix E).

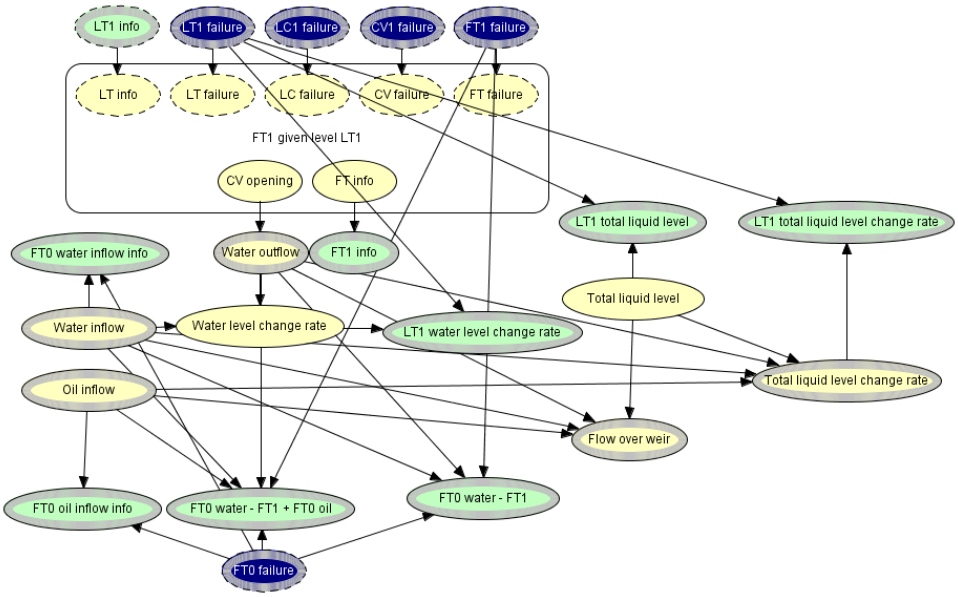


Figure 3.14. BBN *"Separation section"* to model liquid propagation in separation section

3.4.3.4 BBN model for the separated oil section

When building the BBN for the separated oil section some of the generic rules, described in section 3.4.3.2, were not used (specifically Rule 2, Rule 3 and Rule 6), since this section does not have a flow transmitter to measure the inflowing oil from the separation section. No additional section specific rules were necessary. The BBN that was developed for the separated oil section can be seen in Figure 3.15. The details of the BBN nodes can be found in Table 10.3 (see Appendix E).

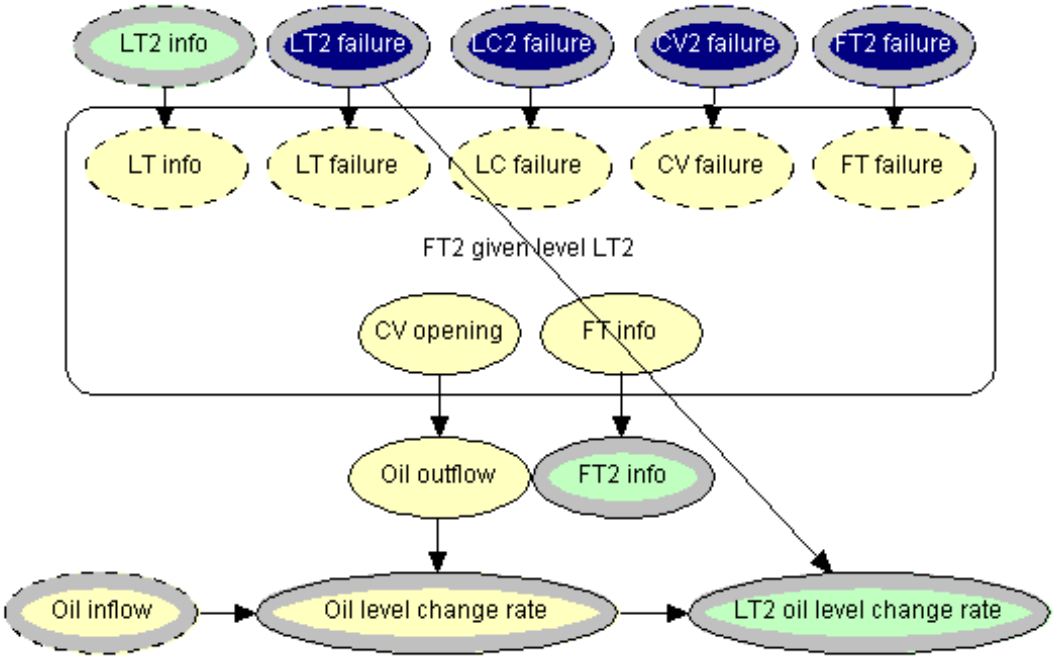


Figure 3.15. BBN *"Separated oil section"* to model liquid propagation in separated oil section

3.4.3.5 BBN model for the gas section

Rule 5 and Rule 7 described in section 3.4.3.2 were not used when building a BBN for the gas section. However, additional nodes had to be created *(*namely *"Change of volume for gas"*, *"Pressure level change rate"*, *"LT3 pressure level change rate"*), since gas is a compressible material and the change of its volume has an effect on a pressure level change rate in the separator. The additional nodes were created in the following order:

Step 1. Node labelled *"Change of volume for gas"* was created to model a change in gas volume in the separator. This node is connected to nodes in BBNs of other section, as will be discussed in section 3.4.3.6. The prior probabilities of all states of this node are equal.

Step 2. Node labelled *"Pressure level change rate"* was created to model pressure changes in the separator based on changes in gas level and volume. Nodes *"Gas level change rate"* and *"Change of volume for gas"* are made parents of this created node. The conditional probability table shows what the expected pressure level change rate is for the given gas level change rate and gas volume change.

Step 3. Node labelled *"LT3 pressure level change rate"* was created to input readings of a pressure level change rate and to model the effects of LT3 failure on these readings. Nodes labelled *"Pressure level change rate"* and *"LT3 failure"* are made parents of this node. The conditional probability table of this node shows how the condition of transmitter LT3 affects the readings of the pressure level change rate.

The BBN that was developed for the gas section can be seen in Figure 3.16. The details of the BBN nodes can be found in Table 10.4 (see Appendix E).

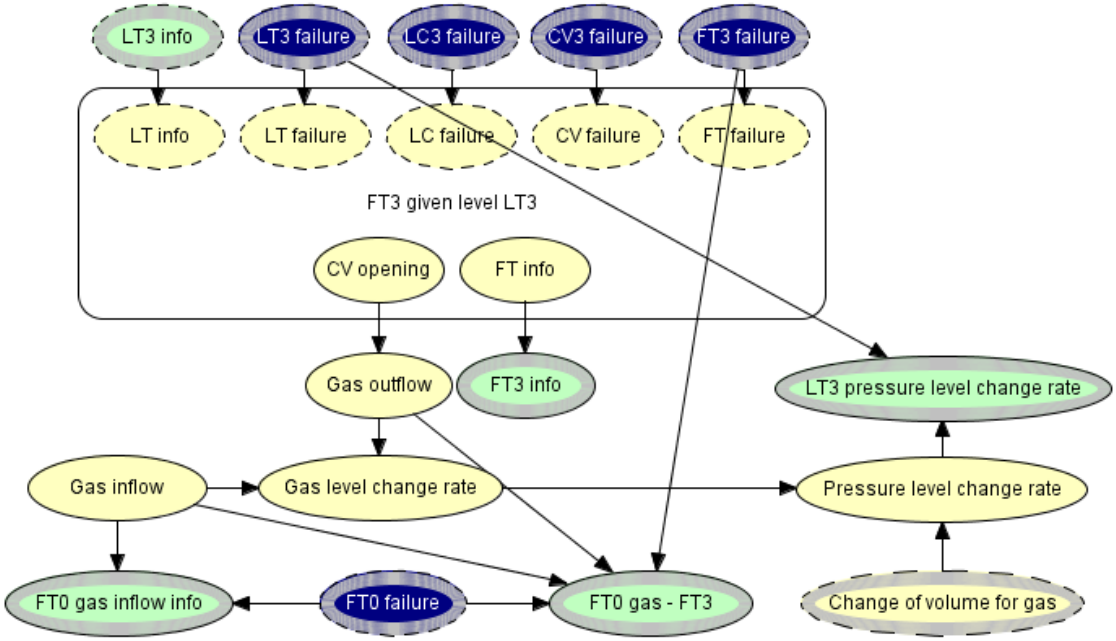


Figure 3.16. BBN *"Gas section"* to model gas propagation in the separator

3.4.3.6 BBN model for interactions between the sections of the three-phase separator

The BBN to model the liquid/vapour propagation in the three-phase separator in a single time slice was developed by combining the individual section BBNs (a separation section (represented by the blue rounded rectangle in Figure 3.17), a separated oil section (represented by the brown rounded rectangle in Figure 3.17) and a gas section (represented by the red rounded rectangle in Figure 3.17)) presented in the previous subsections. The three individual sections were joined by considering how one section can influence flow/level changes of phases in other sections. The following steps were performed to build additional nodes required to model interactions between the three sections:

Step 1. The BBN for the separation section was connected to the BBN for the separated oil section. For that purpose a node labelled *"Flow over weir"* was created, which had identical states to those of the output node labelled *"Flow over weir"* in the separation section BBN. The output node was made parent of the newly created node. The conditional probability table was filled with ones along the diagonal for the created node. This created node was then made a parent node of the input node labelled *"Oil inflow"* in the separated oil section BBN.

Step 2. A node labelled *"Total liquid level change rate single"* was created to obtain the output from the separation section BBN node with the same label (except for *"single"*) and was made a child of the output node. The conditional probability table was filled with ones along the diagonal for the created node.

Step 3. A node labelled *"Oil level change rate single"* was created to obtain the output from the separated oil section BBN node with the same label (except for *"single"*) and was made a child of the output node. The conditional probability table was filled with ones along the diagonal for the created node.

Step 4. Four nodes were created to represent the inflows and outflows of water (*"Water inflow single", "Water outflow single"*) and oil (*"Oil inflow single", "Oil outflow single"*) in order to model the change of volume of gas in the separator. The four nodes were used to get the outputs from the separation section and separated oil section BBNs, thus were made child nodes of the corresponding nodes "Water inflow", "Water outflow", "Oil inflow", "Oil outflow".

Step 5. The BBNs for the separation and the separated oil sections were connected to the BBN for the gas section by creating a new node *"Change of volume for gas from FT"*. This node was then made a parent of the input node labelled *"Change of volume for gas"* in the gas section BBN. At the same time, this newly created node was made a child node of nodes created in Step 4 and labelled *"Water inflow single", "Water outflow single"*, *"Oil inflow single", "Oil outflow single"*. The conditional probability table of the newly created node shows how the gas volume changes in the separator for given inflows and outflows of the water and oil.

Step 6. An output node labelled *"FT0 water – FT1 + FT0 oil – FT2"* was created It is used to input the information on how the liquid levels change in the separator using readings of the flow transmitters showing changes in gas volume. The conditional probability of this node shows what readings will be shown for the given water and oil inflows and outflows and conditions of flow transmitters FT0, FT1 and FT2.

Step 7. An output node labelled *"LT1 total liquid level change + LT2 level change"* was created. It is used to input the information on how the liquid levels in the separator change using readings of the level transmitters showing changes in gas volume. The conditional probability of this node indicates what readings will be shown by transmitters given the actual changes of in gas volume and conditions of level transmitters LT1 and LT2.

Step 8. Nodes labelled *"FT0 varied"*, *"FT1 varied"*, *"FT2 varied"* and *"FT3 varied"* were created to indicate whether readings of the corresponding transmitters have changed through a considered sampling interval. They were made as child nodes to nodes labelled *"FT0 failure"*, *"FT1 failure"*, *"FT2 failure"* and *"FT3 failure"* respectively.

Step 9. The remaining nodes created for this BBN are identical to those input and output nodes of the three sections created previously (except for nodes labelled *"Flow over weir"*, *"Total liquid level change rate"*, *"Oil level change rate"* and *"Change of volume for gas from FT"* which were created in Step 1, Step 2, Step 3 and Step 5).

The BBN that was developed for the interactions between the sections of the three-phase separator can be seen in Figure 3.17. The details of the BBN nodes can be found in Table 10.5 (see). Multiple instances of the BBN developed in this section were then used to build the final OOBN model.

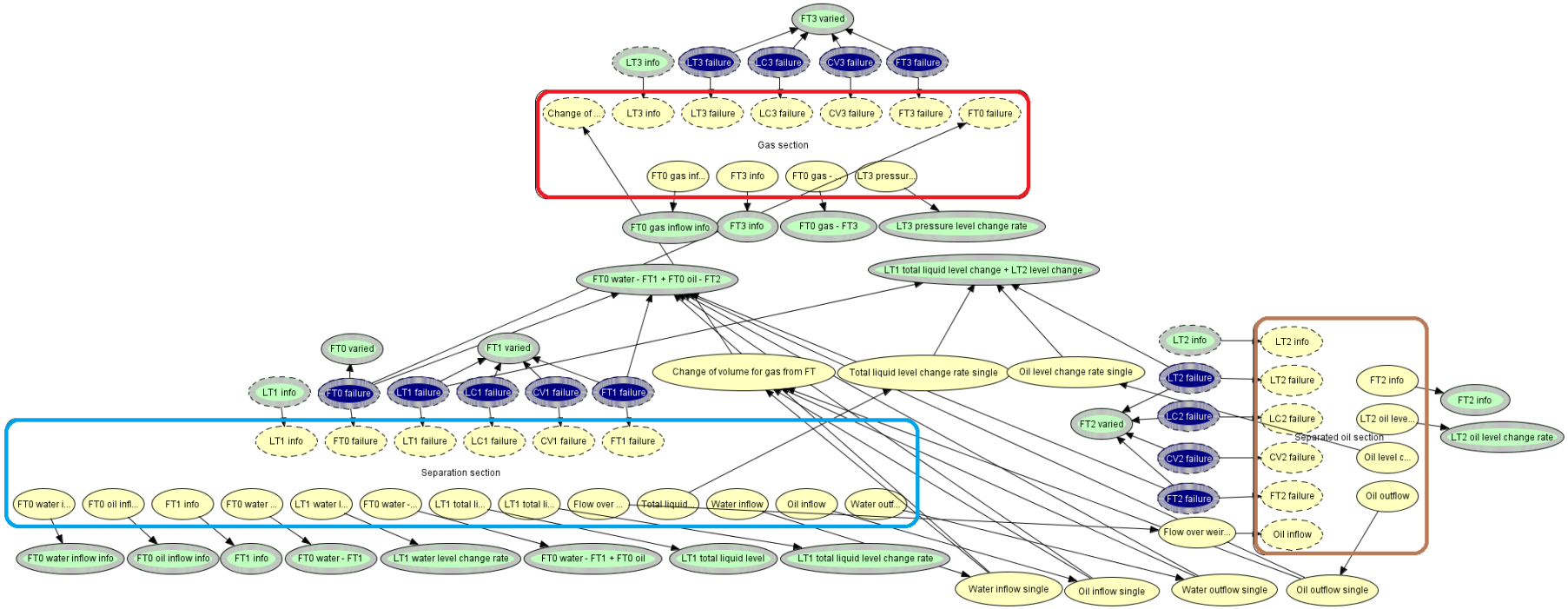


Figure 3.17. BBN *"Single time slice"* for interactions between the sections of three-phase separator

3.4.3.7 Triple time slice OOBN for fault detection and diagnostics of the three-phase separator

The fault detection and diagnostics OOBN model was developed by combining three instances of the *"Single time slice"* BBN introduced in section 3.4.3.6. Three *"Single time slice"* instances were combined, by making the condition nodes from a previous time slice parents of the same nodes in the next time slice, as shown in Figure 3.19. This way, if a failure is detected during one of the time slices, sensor readings in the next slices can be used to validate whether the sensor readings actually correspond to a failure or not. The information nodes (coloured in pale green in Figure 3.19) are used to input sensor readings into the OOBN model. The condition nodes (coloured in dark blue in Figure 3.19) are used to get the output of the OOBN model.

The flowchart given in Figure 3.18, demonstrating how conditions of the components of the three-phase separator are monitored throughout the whole operational time is summarised in the following steps:

Step 1. Enter the pre-processed sensor readings into the OOBN. If any sensor readings are present in the OOBN, shift sensor readings in information nodes (nodes coloured in pale green in Figure 3.19) back by one time slice when new sensor readings become available. For example, sensor readings entered in information node *"FT1 info t-1=>t"* are shifted to information node *"FT1 info t-2=>t-1"*.

Step 2. Update the posterior probabilities throughout the OOBN given new sensor readings.

Step 3. Track how the posterior probabilities of the condition nodes in the most recent time slice changes (nodes coloured in dark blue in Figure 3.19 with a suffix *"t-1=>t"*) in order to detect and identify failures of components. A certain threshold has to be set for posterior probabilities so that failure modes could be detected and identified.

Step 4. Check if posterior probabilities of the condition nodes differ from prior probabilities of the same condition nodes in adjacent time slices. A certain threshold can be set in order to ignore very small differences between the prior and posterior probabilities. For example, if the posterior probabilities of condition node *"FT0 failure t-2=>t-1"* differ only slightly from the prior probabilities of condition node *"FT0 failure t-3=>t-2"*, the difference is ignored.

Step 5. If the difference between posterior and prior probabilities found in Step 4 is significant, make posterior probabilities of the condition nodes as prior probabilities of the condition nodes in a previous time slice For example, the posterior probabilities of condition node *"FT0 failure t-2=>t-1"* are set as prior probabilities of condition node *"FT0 failure t-3=>t-2"*.

Step 6. Repeat the procedure described in steps 1 to 5 when new sensor readings become available.

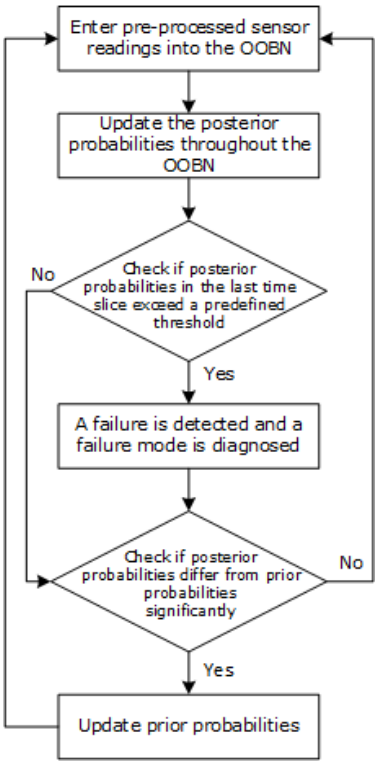


Figure 3.18. A flowchart of fault detection and diagnostics process with OOBN

Such an approach allows conditions of components of the three-phase separator to be monitored once sensor readings become available for all three time slices following the initial start of the monitoring.

Time slices of different lengths can be selected for the analysis and this choice depends on the operation of the separator. For example, if a chosen time slice is too long, by the time the failure is detected it is not possible to prevent the hazardous event. However, if a chosen time slice is too short, no changes in the readings of level transmitters can be observed for a long time due to the operating conditions of the three-phase separator. For the separator considered in this study a time slice of 10 seconds was found to be adequate in order to detect faults in a timely way and observe any changes in the readings of level transmitters when inflows of the liquids or gas differ from their outflows.

Note that the BBNs of "Single time slice" in Figure 3.19 are instance nodes of the *"Triple time slice"* BBN and are labelled as *"t-1 => t", "t-2 => t-1"* and *"t-3 => t-2"*. Time t in this case represents the current time slice of the operation, when sensor readings are available for the whole time slice from the previous time slice $t-1$ (10 seconds earlier in this case). The numbers 1, 2 and 3 indicate the difference of time slices from current time slice t. A higher number of time slices could also be considered.

The number of time slices and the length of a time slice should be considered individually based on the physical properties of the separator and available computing resources. In this study, three time slices were chosen, since it uses readings of 30 seconds of historical data and this was found to be long enough to detect failures of the components, based on the chosen physical properties of the separator.

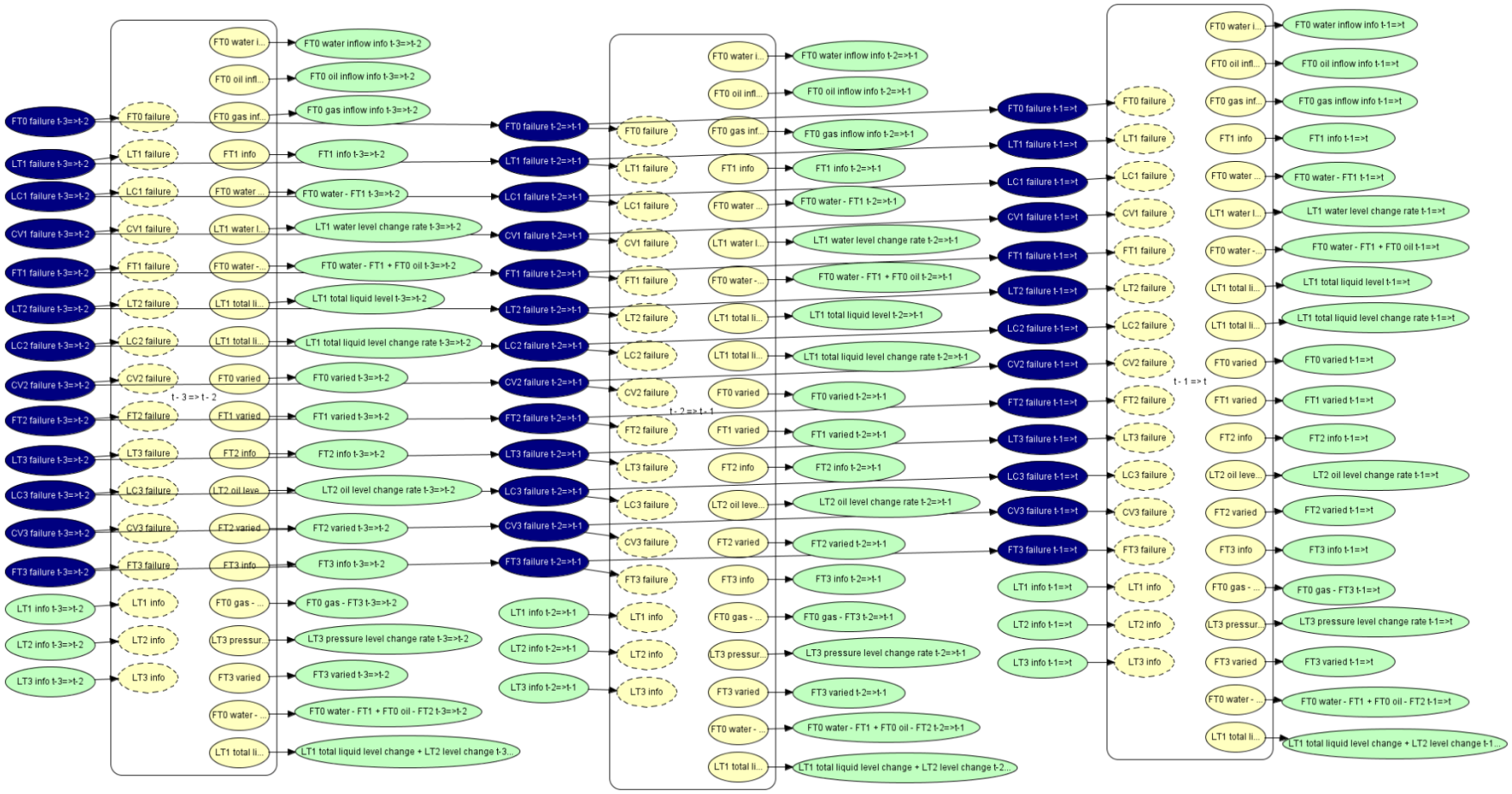


Figure 3.19. *"Triple time slice"* BBN for fault detection and diagnostics of three-phase separator

3.4.4 Summary

A methodology for fault detection and diagnostics of a three-phase separator was presented in this section. Firstly, an overview of the proposed methodology was given. Secondly, the input information required for the fault detection and diagnostics model was introduced. Several combinations of sensor readings have been proposed to be used as input data in order to cross-check the information provided by sensors and to detect the sensor readings that deviate from the expected readings for given conditions. Required pre-processing of input data was also discussed. Pre-processing was performed in order to transform sensor readings into a format suitable for use in the BBN model for fault detection and diagnostics.

Finally, the OOBN model and its constituent parts used for fault detection and diagnostics of three-phase separators were presented. Generic rules for building a BBN model for a separator section were introduced. These rules, together with section specific rules, were used to build a specific section BBN. Then, the BBN to perform the fault detection and diagnostics for the whole separator for a single time step was introduced; it was constructed by joining three individual section BBNs. Multiple BBNs of a single time step were then joined to create a multiple time slice OOBN. The latter is used to perform fault detection and diagnostics by taking into account sensor readings and condition states of the separator components obtained at multiple time instances.

An application of the proposed methodology is presented in the next section, demonstrating the results when single and multiple component failures are introduced in the simulation model of the three-phase separator.

3.5 Application of the proposed methodology using the data of the three-phase separator simulation model

Fault detection and diagnostic capabilities of the proposed methodology are tested by modelling failures in the three-phase separator simulation model and then using the outputs (as detailed in section 3.4.2) from the simulation model as evidence in the OOBN model information nodes to obtain posterior probabilities of component failures in the corresponding condition nodes. An increase in the posterior probability of the failure mode of a particular component in the OOBN will be considered as an indication of the presence of the failure of that component in the simulation model.

Several thresholds for prior and posterior probabilities in the OOBN model had to be set, as discussed in section 3.4.3.7. The threshold of posterior probabilities for failure detection and diagnostics was set to 0.1. The choice of this threshold is arbitrary. It was chosen to reduce the number of false alarms, which might occur if there is a slight increase in the posterior probability, even when no failure is present.

The threshold for the difference of posterior and prior probabilities of the same node in two adjacent time slices was set as 0.01. If this threshold was not exceeded, no updating of prior probabilities was performed for the particular node.

A number of different failure scenarios were used to test the proposed methodology and the results are presented in this section. All of the considered failure scenarios were simulated with the three-phase separator simulation model first and then produced sensor readings were iteratively inserted as evidence in the OOBN model. Several specific scenarios were chosen to demonstrate both capabilities of the model when dealing with complex and challenging tasks and further advancements needed to improve the capabilities of methodology. For example, testing was performed with oscillating flow, since this type of flow is more dynamic than other flow options and thus it is more similar to the actual operating conditions of a three-phase separator. A more extensive testing of single and multiple failure scenarios was also performed in order to evaluate the performance of the OOBN developed.

3.5.1 Prior probabilities of component failure modes

The prior probabilities of component failure modes used for testing of the methodology are given in Table 3.10.

Table 3.10. Prior probabilities of component failure modes

Failure mode	**Prior probability**
LT1 FS, LT2 FS, LT3 FS	10^{-5}
LC1 FL, LC2 FL, LC3 FL	$5 \cdot 10^{-6}$
LC1 FH, LC2 FH, LC3 FH	$5 \cdot 10^{-6}$
CV1 FL, CV2 FL, CV3 FL	$5 \cdot 10^{-6}$
CV1 FH, CV2 FH, CV3 FH	$5 \cdot 10^{-6}$
FT0 FS, FT1 FS, FT2 FS, FT3 FS	10^{-5}

The prior probability of a working state for the same components was obtained by simply subtracting the prior probabilities of all failure modes from 1. The prior probabilities are usually obtained from historical data or are set based on expert knowledge. The latter option was used in this study.

Results with the selected scenarios are presented in the next sections.

3.5.2 Results from the analysis of specific scenarios

Results of the selected scenarios are presented by plotting the posterior probabilities of the states of condition nodes, which relate to the failure mode considered. Moreover, if additional failures are falsely detected by the OOBN, posterior probabilities of the states of the condition nodes representing these failures are also plotted. The posterior probabilities of a condition node are plotted as a stacked column bar chart, where the posterior probability of a certain state of condition node (representing a certain failure mode of component) is represented by an individual colour. A scenario when a single failure mode of control valve CV2 failing closed is presented next.

3.5.2.1 Scenario 1 – Control valve CV2 failed closed

The first scenario considered is when control valve CV2 fails closed. This failure mode prevents the separated oil from leaving the three-phase separator, leading to overfill of the separated oil section if no preventative actions are taken on time. The failure has been entered after 60 seconds (red circle in Figure 3.20 identifies the failure mode and the time it was inserted) from the start of the simulated operation of the separator. Figure 3.20 and Figure 3.21 show that the OOBN has identified two possible failure modes (CV2 failed closed and LC2 failed low), based on the readings provided. The posterior probability of CV FC failure mode of valve CV2 increased to 0.5, as can be seen in Figure 3.20, at the same time as the failure was inserted in the simulation model. Identical behaviour was observed for controller LC2 (see Figure 3.21). This can be explained by the same effect that these failure modes (LC2 failed low and CV2 failed closed) have on the operation of the separator. In both cases, valve CV2 is fully closed and prevents the oil flow out of the separator. Even though the OOBN cannot isolate the exact failure cause it detects a fault in the separator immediately, i.e. as soon as the information from the sensor readings of the three-phase simulation model with the failure presence is passed on to the OOBN.

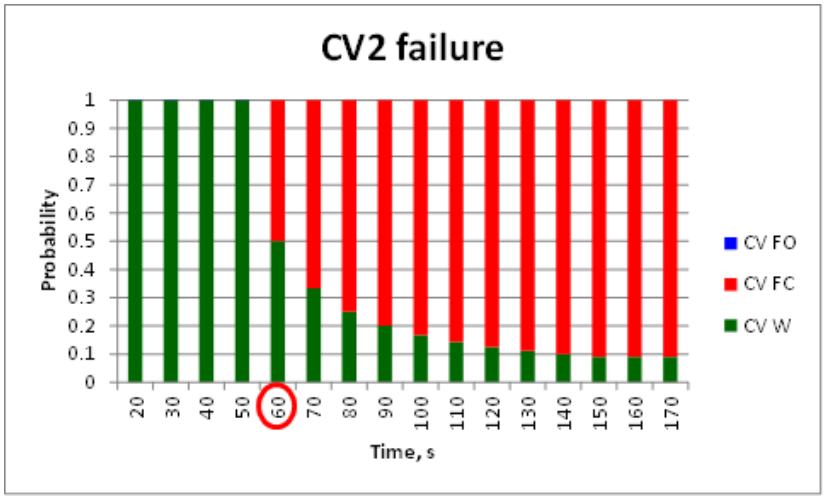


Figure 3.20. Posterior probabilities of the states of the node labelled “CV2 failure t-1 => t”

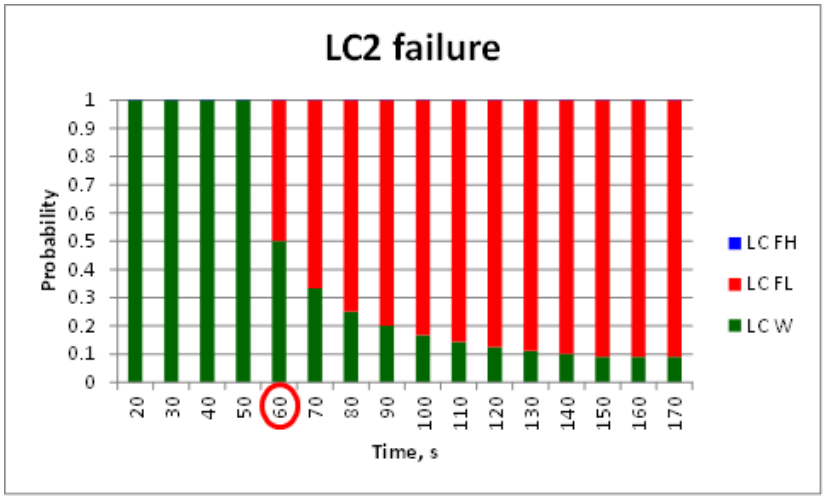


Figure 3.21. Posterior probabilities of the states of the node labelled "LC2 failure t-1 => t"

In the following examples the plots for the level controller failure are omitted, due to the reason that it is always identified as a possible failure mode if a control valve fails and vice versa.

Multiple failures occurring at the same time and with a time lag are presented in the next sections.

3.5.2.2 Scenario 2 – Level transmitter LT1 failed stuck and control valve CV1 failed opened

Failures of level transmitter LT1 failed stuck and control valve CV1 failed opened are considered in this scenario. Both failures are inserted at the same time (60 seconds after the start of the simulation) in the simulation model. The posterior probabilities of failure modes that were identified by the OOBN are given in Figure 3.22 and Figure 3.23. As mentioned previously, the plot for LC1 failure has been omitted.

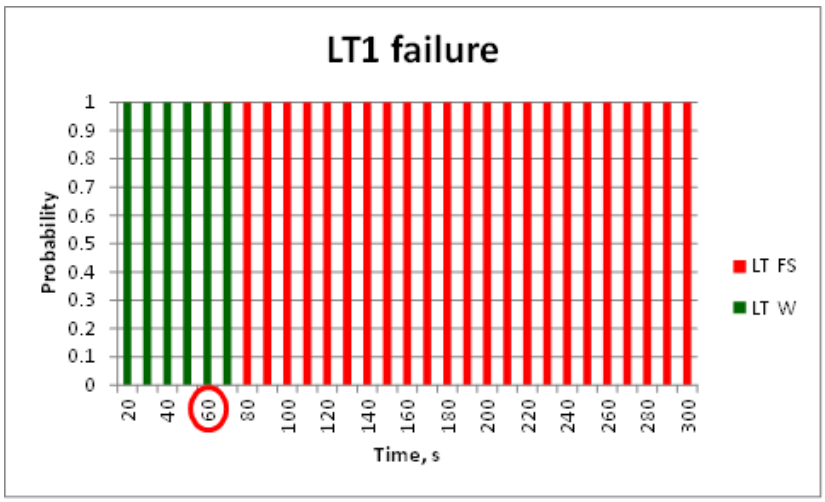


Figure 3.22. Posterior probabilities of the states of the node labelled "LT1 failure t-1 => t"

The failure of LT1 has been identified with a slight delay, as can be seen from Figure 3.22 (failure occurred at 60 seconds, the posterior probability of failure mode LT FS of level transmitter LT1 increased to 1 at 80 seconds). This was due to the fact that

when the failure occurred the water level was above its set point (thus associated valve CV1 was kept open to release the excess of water) and the water inflow was the same as the maximum outflow possible through CV1. For this reason, the actual water level was not changing and therefore no change was expected in the LT1 readings. However, when the water inflow decreased, a change in LT1 reading was expected and thus the failure of LT1 being stuck was identified.

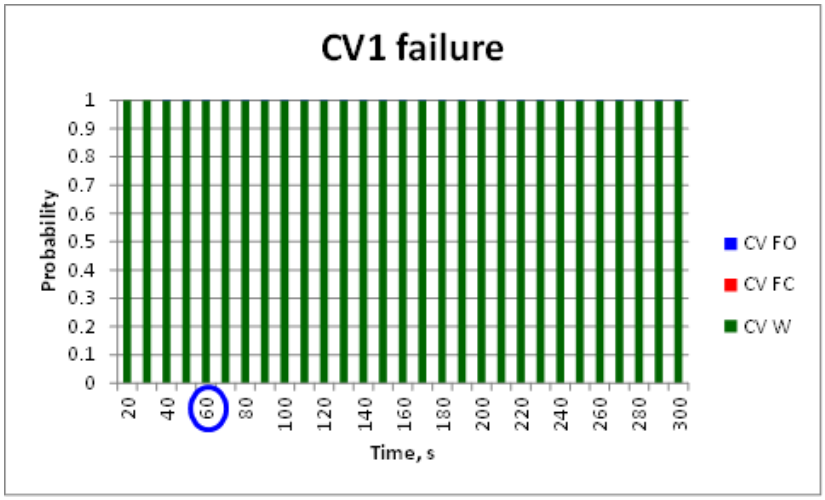


Figure 3.23. Posterior probabilities of the states of the node labelled “CV1 failure t-1 => t”

The failure of CV1 failed open has not been identified as can be seen from Figure 3.23 (the posterior probability of valve CV1 being in state CV W was 1 throughout the testing duration), since the failure of LT1 transmitter forced the valve CV1 to be fully opened and thus maximum flow was expected through CV1. Thus the failure mode of CV1 becomes a hidden failure and can only be detected when the failure of LT1 is rectified.

The same two failure modes are considered next, however in this case the failure of the level transmitter occurs 60 seconds after control valve CV1 fails open. The posterior probabilities for these failure modes can be seen in Figure 3.24 and Figure 3.25.

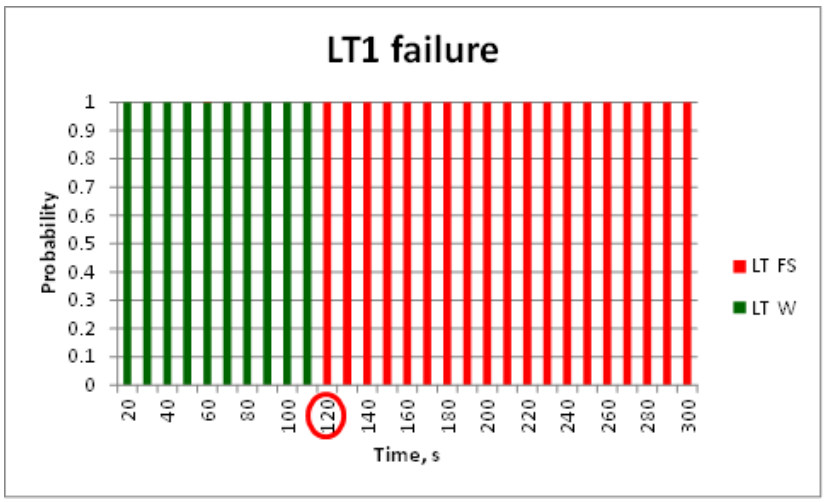


Figure 3.24. Posterior probabilities of the states of the node labelled “LT1 failure t-1 => t”

This time, the failure of LT1 has been identified as soon as it was introduced (see Figure 3.24, red circle is the time of the failure) since the water level was expected to decrease given the water inflow and outflow. Moreover, this time the failure of CV1 failed opened has been detected (posterior probability of failure mode CV FO increased to 0.59 at 90 seconds), as can be seen from Figure 3.25. This can be explained by the fact that level transmitter LT1 sent a water level reading to controller LC1, which indicated that control valve CV1 has to be opened less than the maximum opening of this valve.

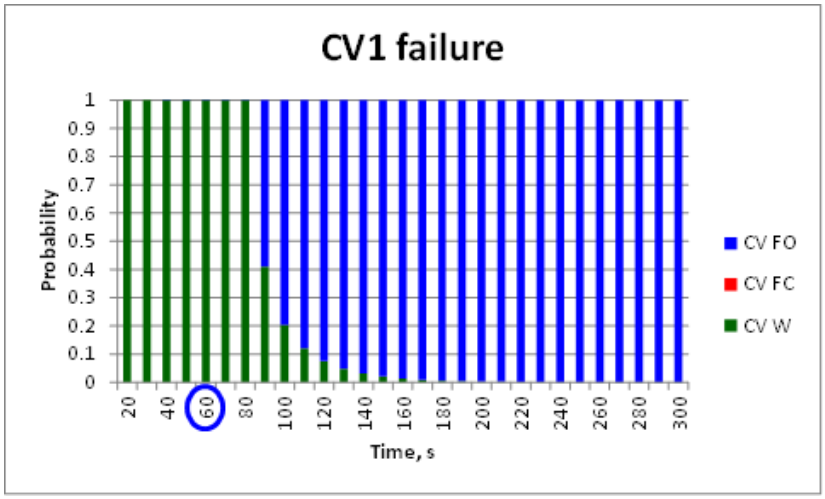


Figure 3.25. Posterior probabilities of the states of the node labelled “CV1 failure t-1 => t”

3.5.2.3 Scenario 3 – Control valve CV1 failed opened and level transmitter LT3 failed stuck

In the third scenario, the failures of CV1 failed open and LT3 failed stuck were inserted in the simulation model (both 60 seconds after the start of the simulation). The posterior probabilities for the states of nodes CV1 failure and LT3 failure calculated by the OOBN after the sensor readings from the simulation model were input into the OOBN are plotted in Figure 3.26 and Figure 3.27.

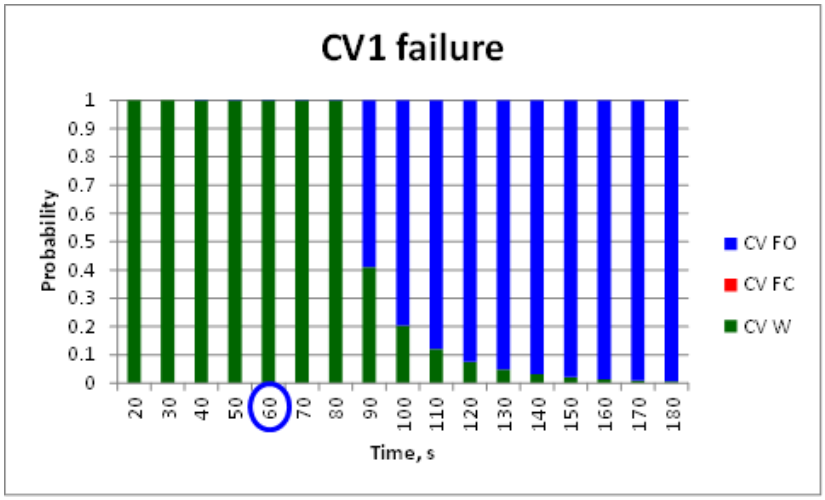


Figure 3.26. Posterior probabilities of the states of the node labelled “CV1 failure t-1 => t”

As previously, the CV1 failed opened failure mode was identified by the OOBN with a time lag if compared to the time when it was inserted in the simulation model. As can be seen from Figure 3.26 failure occurred at 60 seconds and the posterior probability of CV FO increased to 0.59 at 90 seconds. This happened for the same reason as in the previous example, i.e. the water level was above the set point and valve CV1 was expected to be fully opened. As soon as such expectation has changed, the CV1 failed open failure mode was identified.

A similar situation occurred for the LT3 failure. The failure was detected two iterations after it was inserted (failure occurred at 60 seconds, the posterior probability of failure mode LT FS increased to 1 at 80 seconds) as can be seen in Figure 3.27. The reason the OOBN was incapable of detecting the failure at the same time as it occurred was that LT3 was displaying the pressure level as expected with the given inflow and outflow of gas and the changing volume available for gas. As soon as the gas inflow decreased, the failure of LT3 transmitter was identified, since the pressure level in the vessel did not decrease, as was expected.

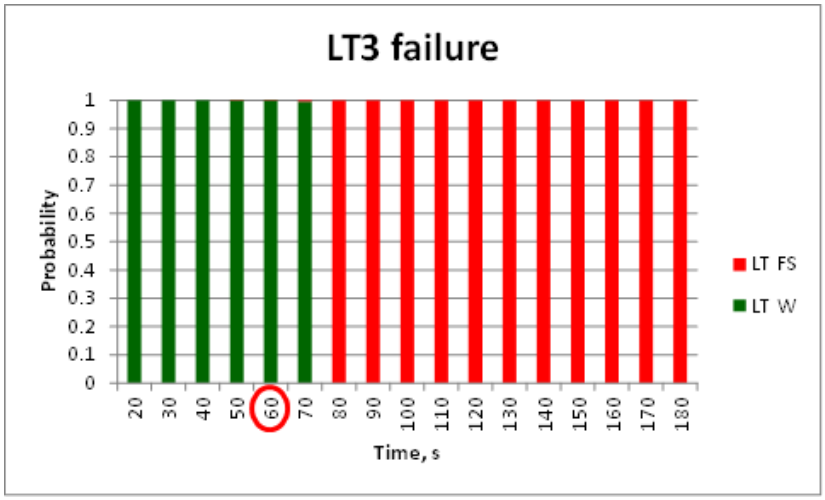


Figure 3.27. Posterior probabilities of the states of the node labelled “LT3 failure t-1 => t”

A summary of the analysis results of a broader choice of scenarios with single and multiple failure modes is presented in the next sections.

3.5.3 Results from the analysis of single faults

All the single failure modes (except for the controller faults, since they have the same effect as failures of the corresponding valve and thus cannot be distinguished from them) were inserted at different time points ($f_t = 30, 40, 50, \ldots, 500$) during the execution of simulation to test the duration of detection lags of the faults given different inflow options. The simulation model was executed for 600 seconds or until an alarm or an undesirable event occurred, whichever was first.

Four different types of inflows were tested in order to validate the proposed methodology on various operating conditions of the three-phase separator. This way, the fault detection and diagnostics is considered to be more challenging, since the separated water, oil and gas levels can change in different ways due to variable inflow rates as well as failure modes of components. Oscillating, decreasing, increasing and random inflows were considered and the results with each type of inflow are presented separately. In total, 624 different scenarios for each type of

inflow modelled were produced to test the OOBN capabilities of fault detection and diagnostics of single failures.

The performance of the proposed methodology is demonstrated by plotting average, minimum and maximum values of lags in failure detection times after the sensor readings from the simulation model were inserted as evidence in the OOBN model. Note that, if a failure mode was not detected by the OOBN model, the duration time of its presence in the simulation model was considered to calculate average, minimum and maximum values of lags.

Additionally, a summary of both detected and undetected failure modes is given. For each correctly detected and diagnosed failure mode its detection and diagnostics potential is determined using the following equation:

$$R_{ID} = \frac{T_d}{T_p} \times 100\%, \qquad \textbf{3.42}$$

where R_{ID} is the ratio of fault detection and diagnostic capability of a specific failure mode ID, T_d is the time duration, when the OOBN was aware that the failure was present, T_p is the time duration the failure was present in the simulation model.

Similarly, the measure of false alarms is given as a ratio expressed using the following equation:

$$R_{F_{ID}} = \frac{T_{F_d}}{T_t} \times 100\%, \qquad \textbf{3.43}$$

where $R_{F_{ID}}$ is the ratio of false alarm of a specific failure mode ID, T_{F_d} is the time duration, when the OOBN was falsely aware about a failure, T_t is the total time of simulation. The results with the oscillating inflows are presented first.

3.5.3.1 Oscillating flow

Oscillating flow was modelled according to the procedure, described in section 3.3.2.1. An example of resulting oscillating water inflow can be seen in Figure 3.28. Oil and gas inflows are modelled in the same way.

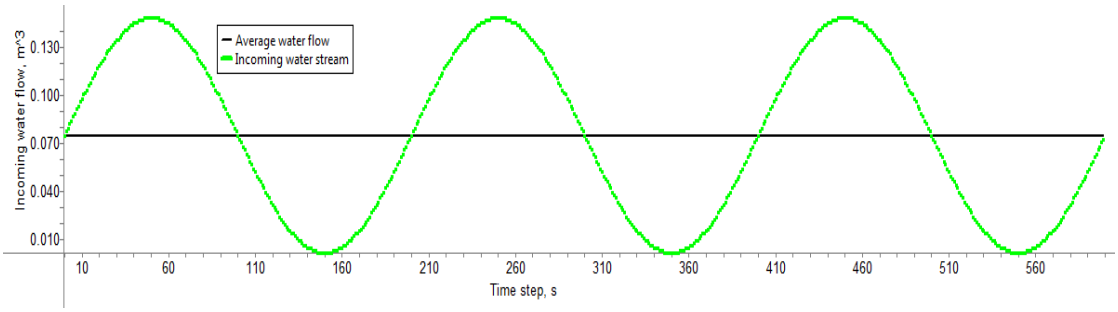


Figure 3.28. Oscillating water inflow as modelled with the three-phase separator simulation model

The main finding of the analysis was that there were no cases where a certain failure mode would not be detected and an undesirable event would occur among all of the 624 scenarios tested. Delays in fault detections using the OOBN model, when oscillating inflows were modelled in the simulation model, are summarised in Figure 3.29. Average, minimum and maximum (blue diamonds, red squares and green

triangles, respectively, in Figure 3.29) values of fault detection delays are presented for each fault.

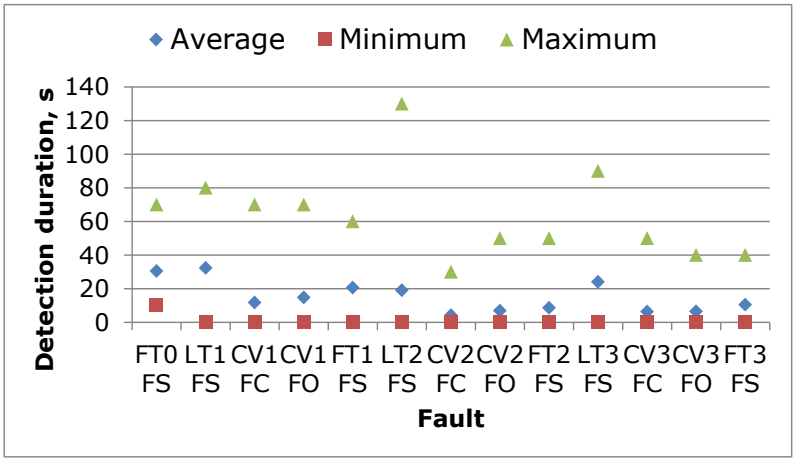


Figure 3.29. Fault detection duration when oscillating inflow was modelled

As one can observe, the average fault detection durations were no longer than 40 seconds, which should give enough time for the operator to perform actions to prevent any undesirable event. The minimum fault detection duration was zero seconds for the majority of faults, except for the flow transmitter FT0 being failed stuck. The maximum fault detection duration was observed for the failure mode level transmitter LT2 being failed stuck. In one case, this failure mode was only detected by the OOBN model 130 seconds after it was inserted into the simulation model. Even though the detection of the failure in this case took longer than the average time, the failure was still detected well before the separated oil level reached the level, where it would raise an alarm if the level transmitter LT2 was not failed. Thus it still gives an earlier indication of a failure than that provided by the alarm methods.

All the failure modes that were detected by the OOBN model are listed in Table 3.11. The first column of the table specifies which failure mode was inserted in the simulation model, while the remaining columns specify the ratios of failure modes identified by the OOBN. When a failure mode in the first column coincided with the same failure mode identified by the OOBN, equation **3.42** was used to determine the ratio of correctly identified failure mode (given as numbers in bold in Table 3.11), otherwise equation **3.43** was used to calculate the ratio of falsely detected failure mode. The same approach to present the results is used in further sections.

As one can observe from Table 3.11, the ratios of time duration when the faults have been correctly detected were very high for the failure modes CV2 FC, FT2 FS, CV3 FC and CV3 FO (ratios higher than 95%). Moreover, for a big number of failure modes tested (9 out of 13) the ratios were higher than 89%. Consequently, in the majority of cases only single faults were detected solely and only very small percentages of false alarms, i.e. additional failure modes were detected. For example, the ratio of false alarms for failure mode LT1 FS was 0.11%, when the FT0 FS failure mode was inserted in the simulation model. The biggest number of false failure alarms (excluding the detection of controller failures when valve failures were inserted and vice versa) was for the gas section components (failure modes LC3 FL, CV3 FC and FT3 FS). These false alarms are raised due to the way the effect of changing gas volume on the change of pressure level is modelled. Finer intervals for these two variables should be considered to achieve a better precision of modelling the change in pressure level and to reduce the number of false alarms.

Table 3.11. Summary of the fault detection and diagnostics ratios, when a single fault has been inserted in the simulation model, when oscillating inflows were modelled

Fault inserted	**Fault detected**																		
	FT0 FS	LT1 FS	LC1 FH	LC1 FL	CV1 FC	CV1 FO	FT1 FS	LT2 FS	LC2 FH	LC2 FL	CV2 FC	CV2 FO	FT2 FS	LT3 FS	LC3 FH	LC3 FL	CV3 FC	CV3 FO	FT3 FS
FT0 FS	**77.11**	0.11																	
LT1 FS		**81.79**								0.64	0.64		0.64						
CV1 FC				29.32	**90.12**										0.40	2.87	2.47	0.40	3.21
CV1 FO			33.60			**89.89**									0.32			0.32	1.06
FT1 FS			1.40	1.26	1.26	1.40	**74.32**												
LT2 FS		0.17						**84.50**							1.01	3.69	2.68	1.01	3.19
CV2 FC										25.30	**95.29**					0.87	0.87		0.94
CV2 FO									20.46			**90.41**							
FT2 FS									0.22	0.25	0.25	0.22	**96.64**						
LT3 FS														**92.66**					
CV3 FC																42.43	**96.74**		
CV3 FO															41.27			**96.47**	
FT3 FS															1.33	1.11	1.29	1.15	**93.91**

3.5.3.2 Decreasing flow

Decreasing flow was modelled according to the procedure, described in section 3.3.2.2. An example of resulting decreasing water inflow can be seen in Figure 3.30. Oil and gas inflows are modelled in the same way.

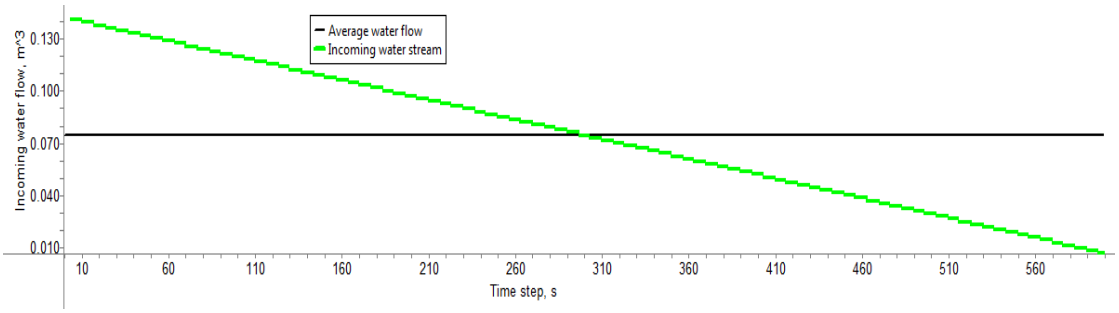


Figure 3.30. Decreasing water inflow as modelled with a three-phase separator simulation model

The fault detection delays when decreasing inflows were modelled are summarised next. The results for each failure mode are given in Figure 3.31. The same graphical representation is used as in the previous section with the oscillating flow.

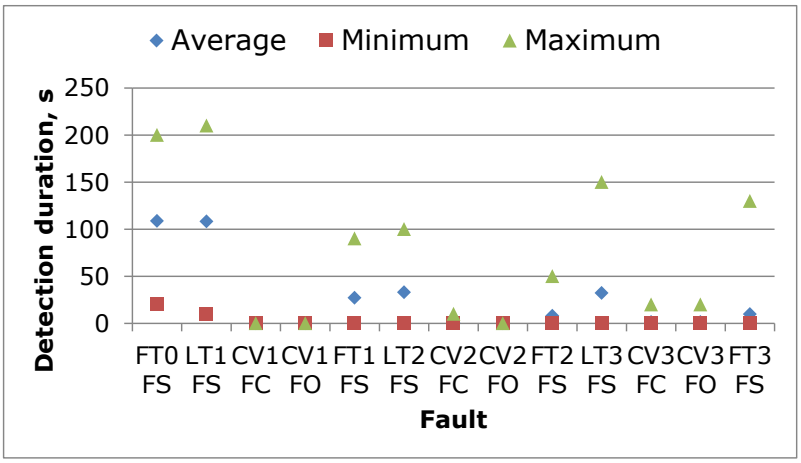


Figure 3.31. Fault detection duration when decreasing inflow was modelled

The majority of average fault detection durations once again were reasonably short, with less than 50 seconds of average detection lag. However, this time the average fault detection durations have increased significantly for failure modes FT0 FS and LT1 FS if compared to the case when the oscillating flow was modelled, i.e. the average duration of detection of FT0 FS was 30.42 seconds and 32.29 seconds for LT1 FS for oscillating flow while for decreasing flow these were 108.75 and 108.13 seconds respectively. The maximum fault detection duration was observed for level transmitter LT1 being failed stuck and failure transmitter FT0 being failed stuck. In some cases, the latter failure mode was only detected by the OOBN model 210 seconds after it was inserted into the simulation model. The delayed detection of the FT0 failure does not impose any risks as the failure of FT0 is not safety critical. In terms of the LT1 fault, even though its detection took longer than the average time, the fault was still detected well before the water level reached the level, where it would raise an alarm if the level transmitter LT1 was not failed.

This time, however, there were a few cases (7 out of 624 scenarios) when no failures were detected even though they were modelled by the simulation model, as can be seen from the summary given in Table 3.12. Even though these failures were not detected by the OOBN, none of them led to an undesirable event or an alarm during the simulation running time. Thus, such failures would not be identified by the traditional method of alarms as well. Components whose failure modes were undetected include flow and level transmitters. The undetected failures of flow transmitters are not safety critical. On the contrary, the undetected failures of level transmitters can be safety critical, thus they are analysed in more detail.

Table 3.12. Summary of undetected failures with OOBN, given decreasing inflows

Fault	FT0 FS	LT1 FS	CV1 FC	CV1 FO	FT1 FS	LT2 FS	CV2 FC	CV2 FO	FT2 FS	LT3 FS	CV3 FC	CV3 FO	FT3 FS
Times **Fault** not detected	1	2	0	0	0	0	0	0	0	1	0	0	3

The LT1 failure mode was undetected when it was inserted at 470 and 480 seconds after the start of the simulation. In both situations, the water level shown by LT1 was equal to the set point when the failures occurred and the actual water level fluctuated around the set point afterwards until the end of the simulation. Thus the shown water level was as expected given the inflow and outflow of water and therefore the fault could not be detected. If the operating conditions of the three-phase separator changed significantly (for example, the inflow of water started increasing, this failure mode would be detected with a big time lag, as seen from the similar scenarios analysed in this study.

The LT3 failure mode was undetected when it was inserted at 450 seconds after the start of the simulation model. The failure mode was undetected because there was only a slight difference between the gas inflow and outflow and the volume available for the gas was changing compensating the difference between inflow and outflow of gas modelled in the OOBN. Moreover, the true pressure level never decreased or increased significantly to the level that would cause an undesirable event or raise an alarm if the level transmitter LT3 was working. Thus, given the operating conditions of the three-phase separator, this failure mode becomes hidden, since the effects of the failure mode are similar to the ones if the LT3 transmitter is working.

As one can observe from Table 3.13, the ratios of time duration when the faults have been correctly detected were very high for the failure modes of valves CV1, CV2 and CV3 (ratios higher than 99%). Moreover, for the majority of failure modes tested (10 out of 13) the ratios were higher than 90%. The ratios of time duration when the faults have been correctly detected have decreased for certain failure modes and have increased for other failure modes. A notable increase can be observed for the failure mode FT1 failed stuck. When the decreasing inflows were modelled, the failure mode was correctly identified by the OOBN 91.92% of the time while it was present in the simulation model. When the oscillating inflows were modelled this ratio was 74.32%. However, notable decreases were observed for failure modes FT0 failed stuck (63.81% for the decreasing inflows and 77.11% for the oscillating inflows) and LT1 failed stuck (63.60% for the decreasing inflows and 81.79% for the oscillating inflows).

Table 3.13. Summary of the fault detection and diagnostics ratios, when a single fault has been inserted in the simulation model, when decreasing inflows were modelled

Fault inserted	**Fault detected**																		
	FT0 FS	LT1 FS	LC1 FH	LC1 FL	CV1 FC	CV1 FO	FT1 FS	LT2 FS	LC2 FH	LC2 FL	CV2 FC	CV2 FO	FT2 FS	LT3 FS	LC3 FH	LC3 FL	CV3 FC	CV3 FO	FT3 FS
FT0 FS	**63.81**	1.83																	
LT1 FS		**63.60**						0.31											
CV1 FC				43.35	**100.00**			0.14											
CV1 FO			39.22			**100.00**													
FT1 FS							**91.92**												
LT2 FS		1.19						**78.31**						0.72	0.10	0.05	0.05	0.05	0.26
CV2 FC										29.22	**99.79**		0.18	0.18		0.30	0.30		0.30
CV2 FO		0.13		0.13	0.13		0.19		24.13			**100.00**	0.06	0.45	0.26	0.13	0.13	0.13	0.39
FT2 FS									0.32	0.07	0.14	0.29	**97.01**						
LT3 FS														**90.16**					
CV3 FC																43.82	**99.35**		
CV3 FO															42.14			**99.31**	0.05
FT3 FS		0.04		0.04	0.04		0.04								2.33	1.44	1.80	1.90	**93.59**

3.5.3.3 Increasing flow

Increasing flow was modelled according to the procedure described in section 3.3.2.2. An example of the result of increasing water inflow can be seen in Figure 3.32. Oil and gas inflows are modelled in the same way.

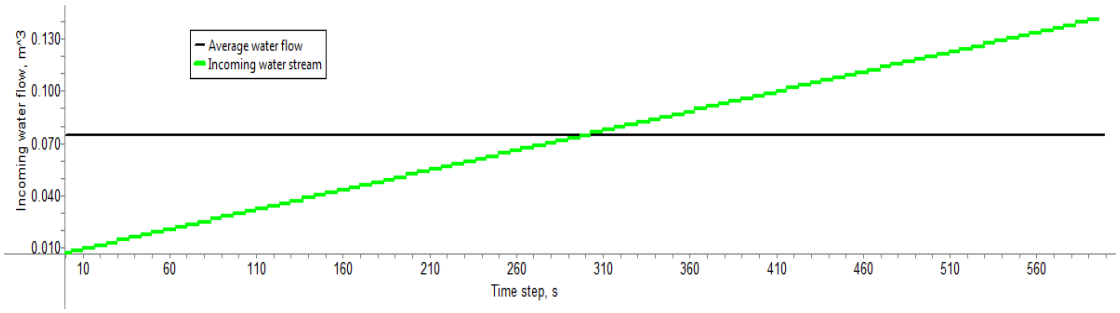


Figure 3.32. Increasing water inflow as modelled with a three-phase separator simulation model

The fault detection times when increasing inflows were modelled, are summarised in Figure 3.33.

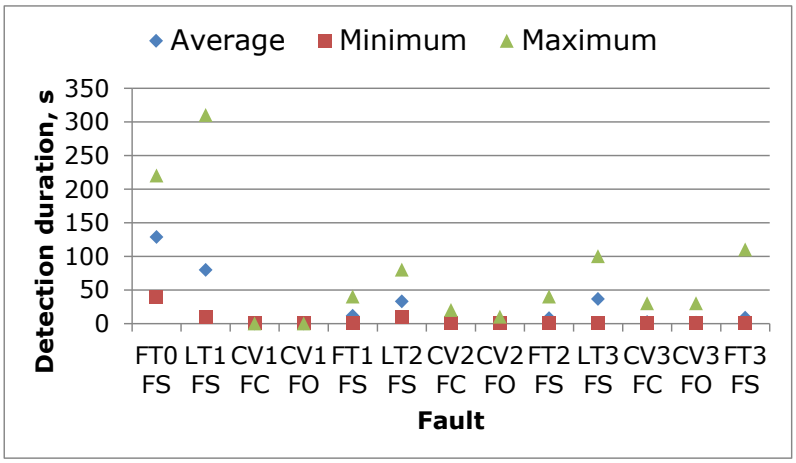


Figure 3.33. Fault detection duration when increasing inflow was modelled

The average values of fault detection times once more were below 50 seconds for the majority of failure modes. They have only increased significantly, compared to the case when the oscillating flow was modelled, for failure modes FT0 FS and LT1 FS, i.e. the average time duration of detection of FT0 FS was 30.42 seconds and LT1 FS 32.29 seconds for the oscillating flow case while for increasing flow case these were 128.54 and 79.79 seconds respectively. The maximum fault detection delay was observed for the level transmitter LT1 being failed stuck failure mode. In one simulation this failure mode has only been detected by the OOBN model 310 seconds after it was inserted into the simulation model. However, even though the detection of the failure in this case took a significantly longer time than the average detection time, the failure was still detected well before the water level reached the level, where it would raise an alarm if the level transmitter LT1 was working.

As with the decreasing flow scenario the failures were detected in the majority of tested simulations (616 out of 624 successful fault detections) with a few cases when no failures were detected (8 out of 624 scenarios, see Table 3.14) even though they were inserted in the simulation model. Despite the fact that these failures were not

detected by the OOBN model, none of them led to an undesirable event or an alarm during the simulation period. Furthermore, some undetected failures such as the failures of flow transmitters are not safety critical. On the contrary, undetected level transmitter failures can be safety critical, thus they are analysed in more detail.

Table 3.14. Summary of undetected failures with OOBN, given increasing inflows

Fault	FT0 FS	LT1 FS	CV1 FC	CV1 FO	FT1 FS	LT2 FS	CV2 FC	CV2 FO	FT2 FS	LT3 FS	CV3 FC	CV3 FO	FT3 FS
Times **Fault** not detected	1	4	0	0	0	0	0	0	0	1	0	0	2

The LT1 failure mode was undetected when it was introduced in the simulation model at 310, 320, 400 and 480 seconds after the start of the simulation. In all the situations the water level shown by LT1 was equal to the set point (the desired level) when the failures occurred and the actual water level did not deviate significantly from the desired level until the end of the simulation. Thus, the shown water level was as expected given the inflow and outflow of water. Moreover, the oil level in the separated section was not significantly affected as well as the pressure level. For this reason the failure of LT1 remained undetected.

The LT3 failure mode was undetected when it was introduced in the simulation model at 500 seconds after the start of the simulation. The failure mode was undetected because there was only a slight difference between the gas inflow and outflow. Moreover, the volume available for the gas was changing compensating for the difference between inflow and outflow of gas modelled in the OOBN. Moreover, the true pressure level never decreased or increased significantly so it could cause an undesirable event or raise an alarm if the level transmitter LT3 was working.

As one can observe from Table 3.15, the ratios of time duration when the faults have been correctly detected were very high for the failure modes of valves CV1, CV2 and CV3 (ratios higher than 98%) and flow transmitters FT1, FT2 and FT3 (ratios higher than 96%). Moreover, for the majority of failure modes tested (9 out of 13) the ratios were higher than 96%. The failure modes FT0 failed stuck and LT1 failed stuck proved most difficult to identify using the OOBN, as was the case with the decreasing flows, as the results in Table 3.15 suggest. However, uniquely to the increasing inflow case, there were a lot of false alarms of LT2 failed stuck (ratio up to 5.17%), when other failures were actually present in the separator. This was due to the fact that towards the end of the simulation the oil level was increasing slowly and due to the transmitter precision modelled the increase was not visible in the LT2 transmitter readings, while the flow transmitters FT0 and FT2 indicated that the level shown by LT2 should be increasing as well.

Table 3.15. Summary of the fault detection and diagnostics ratios, when a single fault has been inserted in the simulation model, when increasing inflows were modelled

Fault inserted	**Fault detected**																		
	FT0 FS	LT1 FS	LC1 FH	LC1 FL	CV1 FC	CV1 FO	FT1 FS	LT2 FS	LC2 FH	LC2 FL	CV2 FC	CV2 FO	FT2 FS	LT3 FS	LC3 FH	LC3 FL	CV3 FC	CV3 FO	FT3 FS
FT0 FS	**60.01**							3.95											
LT1 FS		**69.91**						1.80											
CV1 FC				41.05	**100.00**			0.10											
CV1 FO			41.23			**100.00**									0.10	0.05	0.05	0.05	0.50
FT1 FS							**96.58**	5.17											
LT2 FS								**79.92**							0.20	0.41	0.20	0.20	0.41
CV2 FC		0.06								27.56	**99.12**				0.06	0.18	0.12	0.06	0.25
CV2 FO									25.65			**99.75**	0.13		0.32	0.13	0.13	0.19	0.38
FT2 FS								3.48	0.07	0.18	0.18	0.07	**97.08**						
LT3 FS								4.18						**88.34**					
CV3 FC								1.22								41.95	**98.85**		
CV3 FO								3.42							43.82			**99.35**	
FT3 FS								5.17							1.01	0.57	0.43	1.15	**96.33**

3.5.3.4 Random flow

Random flow was modelled according to the procedure described in section 3.3.2.3. An example of resulting random water inflow can be seen in Figure 3.32. Oil and gas inflows are modelled in the same way.

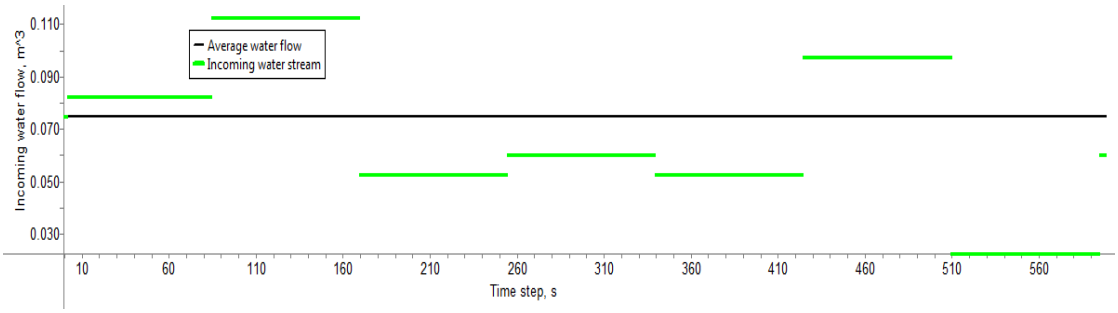


Figure 3.34. Increasing water inflow as modelled with a three-phase separator simulation model

The fault detection times when random inflows were modelled, are summarised in Figure 3.35.

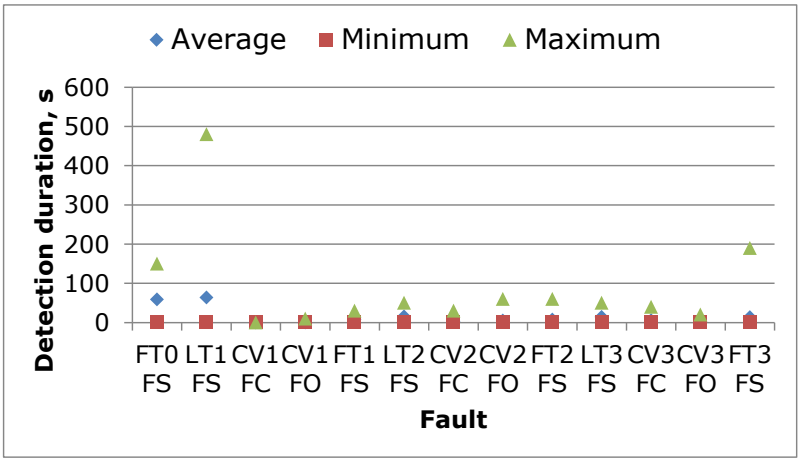


Figure 3.35. Fault detection duration when random inflow was modelled

The average fault detection durations when using the OOBN model have increased if compared to the case when the oscillating flow was modelled only for the failure modes FT0 FS and LT1 FS. The increase, however, was not as drastic as with decreasing and increasing inflows, i.e. the average duration of detection of FT0 FS was 30.42 seconds and LT1 FS 32.29 seconds for oscillating flow while for random flow these were equal to 58.75 and 63.96 seconds respectively. The maximum delay in the fault detection was observed for level transmitter LT1 being failed stuck. In one simulation this failure mode has only been detected by the OOBN model 480 seconds after it was inserted into the simulation model. Even though the detection of the failure in this case took a significantly longer time than the average detection time, the failure was still detected well before the water level reached a level, where it would raise an alarm if the level transmitter LT1 was working.

As with the decreasing and increasing flows the failures were detected in the majority of tested simulations (622 out of 624 successful fault detections) with a few cases when no failures were detected (2 out of 624 scenarios, see Table 3.16) even though they were inserted in the simulation model. Specifically, FT3 FS failures were not

detected twice. Even though this failure mode was not detected in several tested scenarios, it is not a safety critical failure mode and it does not affect the operation of three-phase separator in any way, except by indicating incorrect outflow of gas.

Table 3.16. Summary of undetected failures with OOBN, given increasing inflows

Fault	FT0 FS	LT1 FS	CV1 FC	CV1 FO	FT1 FS	LT2 FS	CV2 FC	CV2 FO	FT2 FS	LT3 FS	CV3 FC	CV3 FO	FT3 FS
Times **Fault** not detected	0	0	0	0	0	0	0	0	0	0	0	0	2

The results of additionally detected failures are similar as for the other types of inflows modelled. The ratios of time duration when the faults have been correctly detected were very high for the failure modes of valves CV1, CV2 and CV3 (ratios higher than 93%) and flow transmitters FT1, FT2 and FT3 (ratios higher than 91%). Moreover, for the majority of failure modes tested (10 out of 13) the ratios were higher than 91%. The good performance of the OOBN model is also represented by a small amount of false alarms, as can be seen from Table 3.17. Moreover, with the random inflows, only two types of failures, FT0 failed stuck and LT1 failed stuck, were hard to identify (other failures were identified with rates higher than 85%). The identification of the flow transmitter FT0 failure was the biggest issue for the OOBN, when only 67.16% ratio was obtained.

Table 3.17. Summary of the fault detection and diagnostics ratios, when a single fault has been inserted in the simulation model, when random inflows were modelled

Fault inserted	**Faults detected**																		
	FT0 FS	LT1 FS	LC1 FH	LC1 FL	CV1 FC	CV1 FO	FT1 FS	LT2 FS	LC2 FH	LC2 FL	CV2 FC	CV2 FO	FT2 FS	LT3 FS	LC3 FH	LC3 FL	CV3 FC	CV3 FO	FT3 FS
FT0 FS	**67.16**																		
LT1 FS	1.71	**76.55**						1.71						0.08					
CV1 FC	0.18			46.13	**100.00**									0.05					
CV1 FO	0.11		37.46			**99.86**													
FT1 FS	0.07		0.07			0.07	**97.57**												
LT2 FS	2.10							**86.44**											
CV2 FC	0.74									26.48	**96.62**			0.06					
CV2 FO									20.91			**93.24**							
FT2 FS	0.90								0.90	0.90	0.83	1.04	**95.34**						
LT3 FS														**94.65**					
CV3 FC																43.52	**97.17**		
CV3 FO															40.88			**98.80**	
FT3 FS	2.73														0.90	3.99	3.70	1.29	**91.42**

3.5.3.5 Summary

Four different inflow options were modelled using the three-phase separator simulation model in order to test the fault detection and diagnostics performance of the OOBN model with 13 different failure modes. Oscillating inflows were chosen as a very dynamic inflow option. Decreasing/increasing inflows were modelled as slowly changing towards minimum/maximum values of the inflows. Finally, random inflows represented a situation when the inflows changed at certain time points to a new random value.

With the chosen design of the OOBN model (three time steps, each of them of 10 seconds, specific chosen prior probabilities), the model performed best in detecting single failures with oscillating inflows (no single failures were left undetected, average detection times of failures were less than 40 seconds).

Most of the failures were detected immediately. Where there were delays in detection, the average detection lags were no longer than 50 seconds for the majority of failure modes (except for FT0 FS and LT1 FS with decreasing, increasing and random inflows) allowing to give a warning about the failure well in advance of an alarm or undesirable event.

The amount of false alarms produced by the OOBN model was also very small. For example, the maximum ratios of false alarms were 3.69%, 2.33%, 5.17% and 3.99% when oscillating, increasing, decreasing and random inflows were modelled respectively (if excluding the detection of controller failures, when valve failures were inserted, as mentioned previously). Thus the majority of failure modes were detected and diagnosed solely, allowing preventative or corrective actions to be performed on the components that actually need them.

There were a few cases when the failures were left undetected, when decreasing, increasing and random inflows were modelled. However, they were usually not critical for the safe operation (failures of flow transmitters) of the three-phase separator. Moreover, even when some failures were not detected by the OOBN, none of them led to an undesirable event or an alarm during the simulation running time. Usually failures were left undetected, because they were inserted at the end of simulation run and when the true levels of water, oil and gas did not change significantly.

The biggest challenge for the OOBN model was the detection of transmitter failures. This was due to the fact, that the effects of failures depend on the occurrence of the failure and in some cases the failure might be a hidden failure for a certain period of the simulation. This was the reason why failures of transmitters took longer to detect than failures of valves. However, the failure becomes hidden because of specific operating conditions and thus the performance of any of the techniques for fault detection and diagnostics would suffer due to this phenomenon.

Another challenge for the model was slowly changing inflows. In this case the liquid or gas level was also changing slowly. Moreover, if the changes are very small they are treated as non-changing due to the way the transmitter readings are discretised into intervals in the OOBN model.

Thus, the design of the OOBN (the number of time slices in the multiple time slice OOBN, the amount of time points in one time slice etc.) should be considered

carefully to fit the operating conditions of the three-phase separator. The analysis of the detection of multiple failure modes is given in the next section.

3.5.4 Results from the analysis of multiple faults

In this section, the results of the fault detection of multiple failure modes when using the OOBN model are presented. This time, two failure modes occurring in a single simulation run were modelled. The multiple failures have been inserted in the simulation model in the following way: first one of the level transmitter failures (LT1 FS, LT2 FS or LT3 FS) is introduced and followed by one of the remaining possible failures. In total 36 combinations of two failures were obtained for the testing of the OOBN model.

In the subsequent sub-sections, only the results obtained with LT1 FS failure mode are presented, since the same tendencies were observed for the LT2 FS and LT3 FS failures (results are given in Appendix F).

As with the single failure modes, multiple failure modes were inserted at different time points during simulation runs to test the delay of the detection of the faults given different inflow options. The first failure mode was inserted at $f_t = 30, 40, 50, \dots, 500$ seconds, while the second failure mode was inserted with a 30 second time lag after the first one. Once again, the simulation model was executed for 600 seconds or until an alarm, or an undesirable event occurred, whichever was first. In total, 1728 different scenarios were produced for each type of inflow modelled to test the OOBN capabilities of fault detection and diagnostics with multiple failures.

Identically as with the single failures, a failure was considered as detected if the posterior probability representing a certain failure mode in the OOBN was bigger than 0.1. The results are summarised in a similar way as in the analysis of single faults (see section 3.5.3), providing a summary of undetected failures as well as the ratios for the detection of different failure modes considered, based on equations **3.42** and **3.43**. The results with the oscillating inflows are presented first.

3.5.4.1 Oscillating flow

When oscillating inflows were modelled there were no situations when both inserted failures would be undetected. There were only a few cases when the failure of level transmitter LT1 was left undetected (8 times out of 576 simulation runs, as can be seen from Table 3.18. Some of the failure modes which were inserted following the failure mode LT1 FS were detected in the majority of tested scenarios, e.g. the FT0 FS failure mode was detected in all of the 48 tested scenarios. Similar performance was obtained when LT1 FS failure was inserted together with failure modes LT3 FS (undetected in 3 out of 48 scenarios), CV3 FC (undetected in 1 out of 48 scenarios), CV3 FO (undetected in 2 out of 48 scenarios) or FT3 FS (undetected in 1 out of 48 scenarios).

As seen previously when modelling single faults, some failures remained hidden failures for a number of time steps after being inserted in the simulation model and thus could not be detected by the OOBN model instantly. A similar issue becomes apparent when two failure modes are inserted into the simulation model. However, this time, not only conditions of the operation of the separator makes the failures hidden but also the existence of the first failure.

For example, it is highly likely, that if level transmitter LT1 has failed stuck and after some time flow transmitter FT1 fails stuck, the effect of the FT1 failure cannot be seen, since the flow is as expected given the failure and readings of the LT1. This is exactly the situation, which was observed in the majority of tested scenarios with LT1 and FT1 failures, as can be seen from Table 3.18. 48 different scenarios for these two failures were simulated, given oscillating inflows, and in 43 of them the failure mode FT1 FS was left undetected due to the fact that it became a hidden failure.

Another situation when such hidden failures might occur is when valve failures CV1 FC or CV1 FO coincide with the failure of corresponding level transmitter LT1. The effect of these hidden failures can also be seen in Table 3.18 where 19 times out of 48 the CV1 failed closed failure was undetected and 24 times out of 48 the CV1 failed opened failure was undetected.

A fairly high number of FT2 failed stuck failures were also left undetected (11 out of 48). There were two different reasons why these failures were undetected: first one is that after the insertion of the LT1 failure, the simulation model ran only for a few time steps before either an alarm in the three-phase separator was raised or an undesirable event occurred. The second reason was that the FT2 failed while it was showing the expected reading given the oil level in the separated oil section and thus became a hidden failure.

Table 3.18. Summary of undetected failures, when LT1 failed stuck failure was simulated together with another component failing, given oscillating inflows

Fault 1	LT1 FS											
Fault 2	FT0 FS	CV1 FC	CV1 FO	FT1 FS	LT2 FS	CV2 FC	CV2 FO	FT2 FS	LT3 FS	CV3 FC	CV3 FO	FT3 FS
Times **Fault 1** not detected	1	0	0	0	0	2	5	0	0	0	0	0
Times **Fault 2** not detected	0	19	24	43	9	7	4	11	3	1	2	1
Times **Fault 1** and **Fault 2** not detected	0	0	0	0	0	0	0	0	0	0	0	0

Such challenging conditions were also well reflected in the summary of the fault detection and diagnostic ratios as shown in Table 3.19. The failure of LT1 has been detected with at least 68.34% (maximum 88.48% ratio) ratio, when this failure was inserted, while for example the failure of FT1 FS was detected by the OOBN only with 11.32% ratio. Nevertheless, in most cases the second failure detection ratio was above 70% which demonstrates that the model proposed can detect multiple faults (whenever they are not hidden failures) in a timely manner avoiding hazardous consequences.

Table 3.19. Summary of the fault detection and diagnostics ratios, when two faults have been inserted in the simulation model, when oscillating inflows were modelled

Fault 1	Fault 2	Faults detected																	
		FT0 FS	LT1 FS	LC1 FH	LC1 FL	CV1 FC	CV1 FO	FT1 FS	LT2 FS	LC2 FH	LC2 FL	CV2 FC	CV2 FO	FT2 FS	LT3 FS	LC3 FH	LC3 FL	CV3 FC	CV3 FO	FT3 FS
LT1 FS	FT0 FS	**68.03**	**73.33**					0.99	0.10	0.35	0.74	0.39	0.35	0.69		0.30	0.49	0.20	0.30	0.49
	CV1 FC		**82.37**		21.15	**72.08**					1.87	1.87		1.87		0.32	0.64	0.32	0.32	0.96
	CV1 FO		**88.48**	22.08			**51.95**		0.35							0.26			0.26	0.87
	FT1 FS		**81.79**	0.20	0.20	0.20	0.20	**11.32**			0.64	0.64		0.64						
	LT2 FS		**81.49**						**62.50**		2.14	2.09		2.34		1.29	2.14	2.14	1.29	3.64
	CV2 FC		**77.54**								22.66	**75.71**					1.86	1.86		1.86
	CV2 FO		**68.34**							17.67	1.31	1.31	**77.46**	1.43		2.75	0.24	0.24	2.81	2.81
	FT2 FS		**81.79**							0.10	14.36	14.21	0.10	**71.29**						
	LT3 FS		**81.79**												**76.38**					
	CV3 FC		**80.65**													0.10	29.89	**86.91**	0.51	1.32
	CV3 FO		**72.71**								0.23	0.23		0.23		21.85			**89.86**	
	FT3 FS		**81.79**					0.05	0.30						0.05	4.83	7.05	6.07	5.08	**73.55**

3.5.4.2 Decreasing flow

When the decreasing inflows were modelled in the simulation model some of the failure modes were detected in the majority of tested scenarios, when inserted following the fault LT1 FS, e.g. the CV1 FO, CV2 FO and CV3 FO failure modes were detected in all of the 48 tested scenarios. Similar performance was obtained when LT1 FS failure was inserted together with failure modes FT2 FS (undetected in 3 out of 48 scenarios), LT3 FS (undetected in 2 out of 48 scenarios) or CV3 FC (undetected in 1 out of 48 scenarios).

However, overall slightly worse results were produced by the OOBN model if compared to the oscillating inflows. The cases when the failure of level transmitter LT1 was left undetected have increased (85 times out of 576 simulation runs, as can be seen from Table 3.20), when decreasing inflows were modelled. The majority of the cases, when LT1 was left undetected, was observed when this failure mode coincided with failure modes CV2 FC or CV2 FO. Such a situation has occurred due to the fact that the water inflow was slowly decreasing and thus a change in the LT1 readings was not expected. Moreover, when the CV2 FC or CV2 FO failures were inserted, the oil level in the separated oil section quickly reached the level where it produced either a high oil level alarm or a low oil level alarm respectively. Thus, the simulation model ran only for a short period of time after CV2 FC or CV2 FO failures were inserted before the simulation was terminated, which was not long enough for the LT1 failure to be detected. When the simulation model was run longer, the failure LT1 FS was detected after a long delay, given the decreasing inflows. If referred back to single failures (see Figure 3.31), the LT1 FS failure mode was detected with an average time lag of 108.13 seconds and thus in most cases the model did not run for long enough time after the second failure was inserted.

Moreover, there were a few situations when both inserted failures were undetected (4 times out of 576 simulation runs). This happened twice for the FT0 FS failure and FT1 FS failure, when inserted together with the LT1 FS failure. It was due to the fact that the effects of the failures on three-phase separator operation were the same as those when both component work.

Table 3.20. Summary of undetected failures, when LT1 failed stuck failure was simulated together with another component failing, given decreasing inflows

Fault 1	LT1 FS											
Fault 2	FT0 FS	CV1 FC	CV1 FO	FT1 FS	LT2 FS	CV2 FC	CV2 FO	FT2 FS	LT3 FS	CV3 FC	CV3 FO	FT3 FS
Times **Fault 1** not detected	2	2	0	2	10	21	31	2	2	7	4	2
Times **Fault 2** not detected	4	3	0	35	10	5	0	3	2	1	0	3
Times **Fault 1** and **Fault 2** not detected	2	0	0	2	0	0	0	0	0	0	0	0

The results in Table 3.21 show that the second failures were detected quicker than in the case of oscillating flow, where the majority of ratios were above 89%. The amount of undetected LT1 FS failures can also be seen in the summary of the fault detection and diagnostics ratios of the OOBN model, given in Table 3.21. The fault detection ratio of failure LT1 FS ranged from 28.89% (inserted together with CV2 FO failure mode) to 84.82% (inserted together with CV1 FC failure mode). As one can observe, a low number of false alarms was raised by the model with the highest number of LC2 FL and CV2 FC (4.65% of all time points), when the LT1 FS and FT2 FS failure modes were inserted in the simulation model.

Table 3.21. Summary of the fault detection and diagnostics ratios, when two faults have been inserted in the simulation model, when decreasing inflows were modelled

Fault 1	Fault 2	Faults detected																	
		FT0 FS	LT1 FS	LC1 FH	LC1 FL	CV1 FC	CV1 FO	FT1 FS	LT2 FS	LC2 FH	LC2 FL	CV2 FC	CV2 FO	FT2 FS	LT3 FS	LC3 FH	LC3 FL	CV3 FC	CV3 FO	FT3 FS
LT1 FS	FT0 FS	**45.40**	**66.34**						0.23						0.15					
	CV1 FC		**84.82**		44.46	**96.21**		0.24	0.77											
	CV1 FO		**84.15**	39.74			**98.22**	1.08	0.32											
	FT1 FS		**63.88**					**20.20**	0.31											
	LT2 FS		**62.37**						**70.78**						0.19	0.19	0.05	0.05	0.14	0.19
	CV2 FC		**41.90**								27.22	**90.03**		0.05	0.11		0.16	0.16		0.16
	CV2 FO		**28.89**						0.18	20.95			**99.72**	0.18						
	FT2 FS		**63.60**						0.46	0.69	4.65	4.65	0.69	**94.46**						
	LT3 FS		**63.60**						0.31						**89.24**					
	CV3 FC		**52.70**						0.18								40.84	**98.82**		
	CV3 FO		**44.42**						0.38							37.10			**99.24**	0.09
	FT3 FS		**63.60**						0.31							0.54	1.69	1.77	0.54	**94.62**

3.5.4.3 Increasing flow

Similar tendencies to the ones observed for the decreasing inflows have been observed when the increasing inflows were modelled (see Table 3.22). Some of the failure modes were detected in the majority of tested scenarios, when inserted together with LT1 FS, e.g. the CV3 FC failure mode was detected in all of the 48 tested scenarios. Similar performance was obtained when the LT1 FS failure was inserted together with failure modes LT3 FS (undetected in 1 out of 48 scenarios) or CV3 FO (undetected in 1 out of 48 scenarios).

As with the decreasing inflows, the LT1 FS failure was most likely to be undetected when it was followed by CV2 FC and CV2 FO failure modes. Once more, the flows of mixture into the separator were increasing more slowly when compared to the dynamic behaviour of the oscillating flow, thus the water level indicated by LT1 transmitter was expected to remain the same for a longer time. Thus, this should be taken into account when building the OOBN model, if it is known that the inflows to the three-phase separator change slowly.

The highest number of undetected failures was observed for the FT1 FS failure mode. This failure was undetected in 21 out of 48 scenarios. As discussed previously, this is the outcome of the FT1 FS becoming a hidden failure, since it is inserted after the effects of the LT1 FS failure influence the readings of FT1.

As with the decreasing inflows, there were a few situations when both of the failures were undetected. An interesting situation occurred, when the LT1 FS and FT3 FS failure modes were inserted. Since, a failure of LT1 influences the water level in the separation section as well as the flow of oil to the separated oil section, the gas volume changes. The gas pressure level change indicated by transmitter LT3 suggested that given the gas inflow and the change of gas volume, the gas outflow has to be maximum, precisely the reading that FT3 transmitter got stuck at. However, the FT3 reading did not correspond to the LT3 issued command, thus the OOBN model indicated that the failure that occurred was only CV3 FO, instead of two failures LT1 FS and FT3 FS.

Table 3.22. Summary of undetected failures, when LT1 failed stuck failure was simulated together with another component failing, given increasing inflows

Fault 1	LT1 FS											
Fault 2	FT0 FS	CV1 FC	CV1 FO	FT1 FS	LT2 FS	CV2 FC	CV2 FO	FT2 FS	LT3 FS	CV3 FC	CV3 FO	FT3 FS
Times **Fault 1** not detected	5	0	2	4	3	13	12	4	4	8	8	4
Times **Fault 2** not detected	8	5	6	21	10	3	5	7	1	0	1	4
Times **Fault 1** and **Fault 2** not detected	2	0	0	4	0	0	0	0	0	0	0	1

The summary of the fault detection and diagnostics ratios of the OOBN model is given in Table 3.23. Once again, there was only a small number of false alarms, if the detection of a controller failure, when the valve failure was inserted, is not considered. However, the fault detection of certain failures took a long time, due to the reasons discussed previously (hidden failures and slowly changing inflows).

Table 3.23. Summary of the fault detection and diagnostics ratios, when two faults have been inserted in the simulation model, when increasing inflows were modelled

Fault 1	Fault 2	Faults detected																	
		FT0 FS	LT1 FS	LC1 FH	LC1 FL	CV1 FC	CV1 FO	FT1 FS	LT2 FS	LC2 FH	LC2 FL	CV2 FC	CV2 FO	FT2 FS	LT3 FS	LC3 FH	LC3 FL	CV3 FC	CV3 FO	FT3 FS
LT1 FS	FT0 FS	**49.34**	**63.55**						2.20						0.04					
	CV1 FC		**86.99**		34.94	**84.85**		0.22	1.07						0.09					
	CV1 FO		**86.70**	39.94			**91.58**	0.34	3.29											
	FT1 FS		**70.62**		0.08	0.08		**46.50**	1.80											
	LT2 FS		**71.08**						**69.69**						0.13	0.18		0.04	0.18	0.18
	CV2 FC		**58.18**								23.15	**87.74**		0.06			0.17	0.17		0.17
	CV2 FO		**60.06**		0.06	0.06		0.50	0.22	24.21			**91.19**	0.67		0.17			0.17	0.17
	FT2 FS		**69.91**						1.43	0.73	4.29	2.86	1.47	**87.07**						
	LT3 FS		**69.04**						1.83						**84.31**					
	CV3 FC		**61.54**						1.40								35.99	**99.07**		
	CV3 FO		**64.57**						1.34							38.08			**97.86**	
	FT3 FS		**69.91**						1.80							1.51	0.73	0.73	1.51	**93.00**

3.5.4.4 Random flow

Random inflows were the last type of inflows that were modelled in the simulation model. Similarly to the previously tested inflow types, some of the failure modes were detected in the majority of tested scenarios, when inserted together with LT1 FS, e.g. the CV2 FO, LT3 FS, CV3 FC and CV3 FC failure modes were detected in all of the 48 tested scenarios. Moreover, the total amount of scenarios, when both failures were undetected decreased to only 3 out of 576 scenarios, if compared to decreasing and increasing inflows. The failures were undetected in those scenarios, due to difference in actual inflow and outflow of the water being small and thus in all of the OOBN nodes, that used this information, it was considered as being in the same discrete interval.

The biggest challenges for the OOBN model were detecting the failure mode FT1 FS (detected in 20 out of 48 scenarios). However, once again, this failure of FT1 becomes a hidden failure in a majority of cases, when it occurs after the failure of LT1 FS.

Table 3.24. Summary of undetected failures, when LT1 failed stuck failure was simulated together with another component failing, given random inflows

Fault 1	LT1 FS											
Fault 2	FT0 FS	CV1 FC	CV1 FO	FT1 FS	LT2 FS	CV2 FC	CV2 FO	FT2 FS	LT3 FS	CV3 FC	CV3 FO	FT3 FS
Times **Fault 1** not detected	3	0	0	0	2	8	14	0	0	3	3	0
Times **Fault 2** not detected	19	9	9	28	12	5	0	5	0	0	0	11
Times **Fault 1** and **Fault 2** not detected	3	0	0	0	0	0	0	0	0	0	0	0

The summary of the fault detection and diagnostics ratios obtained with the OOBN model once again suggests that there were only a few false alarms (see Table 3.25). The largest number of false alarms was for the LC3 FL and CV3 FC failure modes, when the failures of LT1 FS and FT3 FS were inserted. As identified in the increasing flow section, failures of LC3 FL and CV3 FC were identified instead of LT1 FS and FT3 FS respectively due to the reading, the FT3 got stuck at and the way the pressure level changed in the separator.

Table 3.25. Summary of the fault detection and diagnostics ratios, when two faults have been inserted in the simulation model, when random inflows were modelled

Fault 1	Fault 2	Faults detected																		
		FT0 FS	LT1 FS	LC1 FH	LC1 FL	CV1 FC	CV1 FO	FT1 FS	LT2 FS	LC2 FH	LC2 FL	CV2 FC	CV2 FO	FT2 FS	LT3 FS	LC3 FH	LC3 FL	CV3 FC	CV3 FO	FT3 FS
LT1 FS	FT0 FS	**31.61**	**80.47**						1.05						0.31					
	CV1 FC	0.45	**89.73**		40.79	**80.52**		0.19							0.04					
	CV1 FO	0.64	**88.31**	33.09			**83.07**	0.46	0.68											
	FT1 FS	1.51	**76.55**					**35.74**	1.05						0.08					
	LT2 FS	2.95	**76.41**						**64.86**						0.09		0.14	0.14		0.14
	CV2 FC	1.90	**65.47**						1.30		22.71	**79.81**		0.05						
	CV2 FO	1.87	**65.95**							26.27			**89.44**		0.05	0.16			0.16	0.16
	FT2 FS	2.95	**77.76**						2.60	6.32	4.65	3.80	6.55	**77.76**	0.08					
	LT3 FS	1.92	**75.83**						1.72						**91.57**					
	CV3 FC	1.96	**73.16**						2.00								38.23	**95.78**		
	CV3 FO	1.39	**71.30**						1.44							36.48			**99.09**	
	FT3 FS	10.43	**78.19**						1.09		0.31	0.08	0.23	0.19	0.39	5.35	10.24	10.20	5.54	**66.40**

3.5.4.5 Summary

Four different types of inflows were modelled in the simulation model in order to test the fault detection abilities of the OOBN model, when multiple faults were present in the separator. Similarly to the single faults case, the multiple faults were inserted at different time points of the simulation. All testing results are given in Appendix F and only a part of them was presented in the previous subsections. The numbers given in this section correspond to the full analysis of the scenarios with multiple failure modes.

The OOBN model for fault detection and diagnostics achieved a high performance, when the oscillating flow was modelled. There were no cases when both inserted failures would be left undetected in all of the 1728 scenarios (see Appendix F for Table 11.1 and Table 11.2). Slightly worse results were obtained for other types of inflows were both failures were undetected in 5 out of 1728, 11 out of 1728 and 3 out of 1728 scenarios for decreasing (see Appendix F for Table 11.4 and Table 11.5), increasing (see Appendix F for Table 11.7 and Table 11.8) and random flows (see Appendix F for Table 11.10 and Table 11.11) respectively. Even though they were undetected through the simulation runtime it was usually due to the fact that another failure was detected (for example, due to the lack of detail the pressure level was modelled the CV3 valve failure was commonly identified instead of flow transmitter failure FT3) or the simulation ran only for a short period of time after the failures were inserted and due to the operating conditions that were present at that time.

Hidden failures due to the operating conditions of the three-phase separator were a major challenge for the timely detection of the failures, when a single failure was inserted. In the case of multiple failures, the number of hidden failures in the tested scenarios just increased. Not only did the failures become hidden because of the operating conditions of the three-phase separator, but also in certain situations an effect of one failure hid the effect of another failure and thus one of the failures was left undetected. This highly influenced the performance of the OOBN model. As a result some of the faults were undetected by the OOBN model in the majority of scenarios (for example FT1 FS was undetected in a lot of scenarios when inserted after the LT1 FS fault).

An issue of multiple failures not being detected in some of the cases was also observed due to the discretisation of inputted information into the OOBN model. When the discretisation of sensor readings was performed the precise information was lost, which led to slight changes in the flow and level transmitter readings being undetected. One of the possible ways to deal with it is to discretise the continuous data into finer intervals. However, these would lead to OOBN nodes having more states and bigger conditional probability tables, which could increase the computation complexity.

3.5.5 Summary of the application of the proposed methodology

The results of the application of the proposed methodology for the fault detection and diagnostics of three-phase separator were presented in this chapter. A methodology chosen for testing the capabilities of the OOBN model was introduced along with prior probabilities used for the OOBN condition nodes. Three specific scenarios were chosen to illustrate how the OOBN model can detect and diagnose certain failure

modes and what are the main challenges for the model. A more extensive testing of the fault detection and diagnostics ability of the OOBN was presented next.

Firstly, only single faults were simulated in the three-phase separator by varying the time points when they occur and the type of inflows to the three-phase separator. Results were expressed as average, minimum and maximum times of the fault detection along with a summary of the undetected faults and the detection and diagnostics ratios of specific failure modes. The OOBN model performed exceptionally well when the oscillating inflows were modelled. No failure modes were left undetected and the average detection times for all of the failure modes were below 40 seconds. Such a fast failure mode detection and identification would allow the operator of the three-phase separator to be aware of a developing hazardous situation in advance and to perform any preventative actions needed. Moreover, the amount of false alarms produced by the OOBN model was also very small. For example, the maximum ratio of false alarms was 3.69%.

For other types of inflows there were a few failure modes left undetected, mainly due to hidden faults, since given certain operating conditions of the three-phase separator some components that failed demonstrated the expected behaviour. Nevertheless, these failure modes were either not safety critical or the operating conditions of the three-phase separator did not deviate significantly from the normal operating conditions throughout the simulation run. In addition, the maximum ratios of false alarms were 2.33%, 5.17% and 3.99% when increasing, decreasing and random inflows were modelled respectively.

Slowly changing operating conditions of the three-phase separator and very small differences between the flow and level transmitter readings, which were lost when discretising the information into the discrete states of the OOBN, were among other challenges for the OOBN model. They influenced the majority of lags in fault detection and diagnostics times, when increasing and decreasing inflows were modelled.

The OOBN model performed slightly worse when multiple failure modes were inserted in the three-phase separator simulation model. The major finding from the analysis of the results with oscillating inflows was that there was no single scenario from all of the 1728 scenarios tested, when both failure modes would be undetected. This is a very promising result, since the OOBN model used for fault detection and diagnostics shows that it can cope with highly varying operating conditions of the three-phase separator.

Slightly worse results were obtained for other types of inflows modelled, where several scenarios existed when both failures were not detected. Nevertheless, they were undetected due to the fact that another failure mode was detected. Thus the model managed to detect a fault and only the type of the fault was identified incorrectly. In other cases, the simulation ran only for a short period of time after the failures were inserted and due to the operating conditions that were present at that time, the failures could not be detected in a timely fashion. They would have been detected if the model would have run for a longer time, as observed for the similar scenarios, where the failure modes have been detected with a bigger time lag.

The issue of hidden faults became even more apparent when multiple faults were inserted in the simulation model. In some cases one of the faults could not be detected due to the fact that the first failure that occurred hid the effects of the second failure. Thus, the end user of the model should be aware of the fact that there might be hidden failures and therefore should interpret model results critically.

3.6 Conclusions

3.6.1 Introduction

The three-phase separator is a critical component of oil and gas processing plants. Effective operation of the three-phase separator ensures that the oil and gas processing plant is working at its maximum capacity and is providing a good quality oil product. Unexpected failures which lead to shutdowns of the three-phase separator or the whole oil and gas processing facility leads to big losses for companies. Moreover, such failures might result in hazardous events, introducing risk to the workforce. Thus techniques to detect and identify failing components as soon as possible are essential.

A common approach used for fault detection is an alarm based system, when certain thresholds are determined and the process is shut down if these are violated. However, such an approach does not allow the timely detection of failures and thus, in the majority of cases, the three-phase separator has to be shut down. Model-based approaches were developed in order to detect faults early, i.e. at their developmental stage. However, the majority of currently-available models cannot cope with multiple failures or are defined for specific system operating conditions and need extensive modifications for the same methodology to be applied to a similar system with different operating conditions.

In this thesis, several novel aspects were proposed for the fault detection and diagnostics of a three-phase separator. The proposed methodology using the OOBN model includes detection and diagnostics of multiple failure modes, which is a major improvement if compared to the previously-proposed approaches. Several other aspects will be overviewed in the following sections.

3.6.2 Bayesian Belief Networks for fault detection of three-phase separator

The methodology for the fault detection and diagnostics of three-phase separator proposed in this thesis is based on the BBN method. An Object Oriented Bayesian Network (OOBN) was built in a modular way:

- Generic rules for building a BBN to model a control loop were proposed. The instances of this BBN were then used to develop BBNs for specific sections.

- Generic rules for building a BBN in a structured way for a section of three-phase separator were proposed. Section-specific rules were introduced to take into account the uniqueness of each section of the three-phase separator. A set of generic and section-specific rules was used to build individual BBNs built for each section (separation section, separated oil section and gas section).

- The BBNs for individual sections were then joined into a single BBN to model operation of three-phase separator during a single time slice (a fixed length time interval). This way the BBN becomes modular and only a corresponding individual section BBN need to be modified if certain operating conditions or physical parameters of one section change.

- A multiple time slice OOBN was proposed by combining multiple instances of the BBN modelling the operation of the three-phase separator during a single

time slice in order to take into account the condition of components in previous time slices.

Two types of specific nodes were introduced in the OOBN to insert input data (information nodes) and get the output (condition nodes) from the model. Sensor readings of a three-phase separator (in this case it was a simulation model) were used as inputs of the information nodes of the OOBN model. Several pre-processing procedures were considered in order to be able to input the continuous data obtained from the separator into the discrete information nodes in the OOBN. Moreover, combinations of transmitter readings were proposed in order to obtain supplementary data about the processes in the separator without a need to install additional equipment. Once the sensor readings from the separator were inserted in the information nodes of the OOBN, the model produced posterior probabilities of the condition nodes states (working state or one of the failure modes of certain component).

The monitoring of the condition of the three-phase separator components was performed by monitoring the posterior probabilities of the condition nodes states in the OOBN model throughout the operation of the three-phase separator, as detailed in section 3.4.3.7, Figure 3.18. Whenever the posterior probabilities in one time slice have changed significantly, these were then set as the prior probabilities of a previous time slice. Such an approach made it possible to keep track of the failures detected by the OOBN earlier. The posterior probabilities in the last time slice of the OOBN were compared to a threshold and whenever it was exceeded the corresponding failure mode was considered to be detected and diagnosed.

3.6.3 Testing the proposed methodology on the simulated three-phase separator

A simulation model of the three-phase separator was built to simulate failures of components of the separator and their effects.

The chosen approach to simulate the operation of a three-phase separator allowed extensive testing of the fault detection and diagnostic abilities of the OOBN model proposed. Four different inflow types were chosen for the testing phase. Single and multiple faults were inserted in the simulation model at different time points. Advantages and disadvantages of the built OOBN model were identified during the testing phase.

The testing of the OOBN model with single failures showed that it can successfully cope with fault detection and diagnostics, especially when oscillating flow was modelled. In that case, failures were detected and diagnosed in all of the 624 scenarios. Moreover, the average detection times were no longer than 40 seconds. Thus, an early warning of one of the component failures can be given to allow the responsible personnel to deal with the developing failure of the separator, before an undesirable event occurs. The maximum ratio of false alarms produced by the OOBN model was 3.69%. Slightly worse results were obtained for other types of modelled inflows, when there were a few cases of undetected failures. Nevertheless, the extensive testing performed with single failures has shown great potential for the developed OOBN model to successfully perform fault detection and diagnostics when using only data available from existing system components.

The testing of the OOBN model with multiple failures was more challenging and thus the obtained results were worse than the ones for the single failures. Nevertheless, the OOBN model successfully detected and diagnosed at least one of the inserted failure modes in all of the 1728 scenarios considered for the oscillating inflows. Slightly worse results were obtained for other inflow options where a few scenarios existed, when both of the inserted failures were not detected by the OOBN model. However, even in those scenarios, the failures were either not safety critical or the operating conditions of the three-phase separator did not deviate significantly from the normal operating conditions throughout the simulation run.

Hidden faults were the biggest challenge, which was observed for both single and multiple failures. In the case of single failures, this occurred when effects of some failures on the operation of the three-phase separator cannot be observed, because given the process conditions the component would perform identically whether it has failed or not. In the case of multiple failures, the effects of some faults on the operation of three-phase separator cannot be observed because another component fault has happened before as well as due to the situations described previously. In order to detect such faults additional sensors would need to be used in the system.

Another issue for the OOBN model was to identify very subtle changes in the operation of the three-phase separator caused by failure of some components, since these subtle changes were usually lost when discretising the sensor readings. This was the main reason for the delay in detection and diagnostics of some of the faults.

These issues should be addressed as part of future developments of the model if it was to be applied in practice.

3.7 Future work

The research performed showed the potential for the Bayesian Belief Network technique to be used as a fault detection and diagnostic tool for three-phase separators. However, further developments of the proposed methodology are necessary in order to be able to use it in practical applications.

There were two major issues identified in the current form of the proposed methodology, which need to be addressed first. Firstly, the issue of not being able to detect slight changes in the operating conditions of the three-phase separator which could lead to undetected failures and hazardous events have to be dealt with. A solution for this problem could be finer intervals of discrete nodes or even the use of continuous nodes to represent readings from flow and level transmitters. This could lead to more accurate fault detection and diagnostics. Secondly, the identification of hidden faults should be addressed. However, this task might be a very hard or near impossible one. Even using finer intervals for discrete nodes or using continuous nodes may not be able to resolve this issue. Additional or redundant sensors could be considered in order to overcome this issue.

The methodology could also be enhanced in order to model the separation process by the BBN. In the study performed in this thesis, the separation process was considered to be perfect and instant, i.e. the liquid and gas flowing into the separator gets separated fully and instantly. The addition of the separation process modelling capability to the BBN would probably make the process of detecting faults more challenging. On the other hand, it should help to identify certain faults. For example, the pressure level has an effect on the speed of the separation process, thus if the separation process is different from what is expected faults related to the pressure control and monitoring loop could be detected with this new feature.

Another important feature that has to be taken into account for the fault detection and diagnostics of the three-phase separator is leak detection. Leaks in different places of the separator might have different effects on the liquid or gas levels in the separator and in some cases might be a critical failure of the separator. Thus, the inclusion of leak modelling in the BBN is recommended.

The proposed methodology is based on monitoring the changes of the liquid/gas in the three-phase separator using sensor readings and does not take into account the actual levels of liquid/gas (except when a PI controller output is calculated). The addition of current levels of liquids/gas and their propagation from one time slice to another could be also included in the methodology.

Other extensions to the proposed methodology could include modelling additional component failure modes (for example valve being stuck half open/closed) or even automatically building a BBN from the specifications of a three-phase separator.

Further applications of the proposed methodology can be investigated. The proposed methodology can also be used to build BBN models for the fault detection of similar systems, for example, two inter-connected separators, free water knockout systems, desalters and other vessel based systems (having similar components: controllers, transmitters, valves) used in the oil and gas industry. It would require slight adjustment of the BBN model to the specific needs of the considered system.

4 Final conclusions

The main objectives that were set out in the beginning of this thesis were fully or partially achieved in the research performed. Each of the objectives is briefly reintroduced next, highlighting the extent to which it was achieved for individual systems.

Propose a methodology for the fault detection and diagnostics of the considered system.

Railway point systems

The methodology for the fault detection of railway point systems based on One Class Support Vector Machines and time series similarity measures was proposed. The choice of the selected methods was done in order to take into the account the features of in-field data of RPS. The fault detection of RPS was considered as a classification task, where the abnormal sensor readings, i.e. POE motor current corresponding to faulty behaviour of RPS, would be isolated from the majority of sensor readings corresponding to the failure free behaviour of RPS. The POE motor currents were compared using time series similarity measures and this information was fed into the OCSVM model to perform the classification between faulty and failure free behaviour of RPS.

However, fault diagnostics for the RPS was not considered in this study. The main reason behind this was the lack of representative data of each failure mode. Moreover, the analysis of system operation under different failure modes present revealed an increase of current used by POE motor for most of the cases. Thus the discrimination between different fault types using only current measurements is impossible in most cases.

Three-phase separators

The methodology for the fault detection and diagnostics of three-phase separators based on Bayesian Belief Networks was proposed in this study. A modular OOBN was built to model changes of water, oil and gas levels in the individual sections and the whole separator when the system is free from faults as well as when some faults exist. The idea of the methodology for fault detection and diagnostics was based on analysing sensor readings from a number of adjacent time intervals of operation of three-phase separator. The posterior probabilities of the states of the nodes representing the condition of components of TPS were monitored throughout the time after sensor readings are inserted in the OOBN model. The increase of the posterior probability representing a certain component failure mode was considered as an indication of that failure present in the system. Thus, the fault detection and diagnostics for the TPS was performed simultaneously.

Implement the proposed methodology in software.

Railway point systems

The software for the fault detection of RPS was developed using a combination of Matlab, C++ and LIBSVM package. Matlab was used to pre-process the data as necessary and provide input to the OCSVM model as well as to obtain output from the OCSVM model and visualise the results. The time series similarity measures were calculated using C++ to take advantage of the speed provided by C++ if compared to Matlab. The OCSVM model was implemented in the LIBSVM package,

which is implemented as Matlab functions to call the C++ engine to perform the calculations necessary and obtain the output in the Matlab format. The software was developed to be as automated as possible, where minimal user input is necessary to perform the analysis and obtain the results from the OCSVM model.

Three-phase separators

Firstly a TPS simulation model was built using C++ and the wxWidgets libraries to create graphical user interfaces. User friendly software with GUI was built in order to allow the user to interactively specify the physical and operational parameters of the TPS as well as the failure modes to be modelled in the selected simulation run. The outputs of each simulation run, that is the sensor readings, were plotted graphically as well as exported to text files for further input to the OOBN model.

The software for fault detection and diagnostics with OOBN was developed using HUGIN graphical user interface (GUI) and application program interface (API) for the C++. BBN models were first built using the GUI of HUGIN and these were later used in the C++ API to perform the fault detection and diagnostics of OOBN. The posterior probabilities of the selected nodes of the OOBN were exported to the text files at selected time intervals as well as the failure modes that exceeded the set thresholds for the fault detection and diagnostics. Visual Basic and Windows scripts were written to automatically compose the results from individual scenarios into a summary of fault detection and diagnostics performance for the selected type of inflow.

Validate the proposed methodology on data representing the operation of considered system.

Railway point systems

The proposed methodology was tested on data of infield RPS. The main difference between in-field and laboratory data was that the data obtained from the laboratory point system to represent the fault-free movements had very little variance. The behaviour of the RPS changed with time and there was a lot of variability even in the non-faulty behaviour of the system. In addition, the operation of RPS was also influenced by severe weather conditions or loads applied to the system which cause this variability. The in-field data showed that some failures cause subtle effects to the behaviour of the system, and the proposed method in this study aimed to detect such effects.

The testing performed in this study was highly affected by the discrepancies present in the data labelling. Results that were achieved with automatic labelling procedures based on alarm or failure dates were very different. A detailed analysis of data labelling results with both labelling techniques was performed which led to a combination of the labelling procedures together with some expert knowledge. The relabelled data was used to test the OCSVM model again and it was shown that the OCSVM model can perform well, as shown in section 2.5.4 and published in (Vileiniskis et al., 2013, Vileiniskis et al., 2015).

Three-phase separators

The proposed methodology was tested on the data obtained from a three-phase separator simulation model. A high number of different scenarios, including single and multiple failures and different inflow options were simulated. This data was then used to perform fault detection and diagnostics with OOBN model. The extensive testing performed with single failures has shown the potential to successfully perform fault detection and diagnostics with the developed OOBN model.

The testing of the OOBN model with multiple failures was more challenging and thus the obtained results were worse than the ones for the single failures. Nevertheless, the OOBN model successfully detected and diagnosed at least one of the inserted failure modes in the majority of all of the scenarios considered (5165 out of 5184 scenarios). Moreover, in the scenarios where both failures were undetected, they never led to undesirable events or liquid/vapour levels where an alarm would be raised if the components were working. The biggest challenge for the OOBN model was hidden failures, which were observed for both single and multiple failures.

Application in practice

Railway point systems

Before applying the methodology in practice, several aspects have to be considered. Firstly, the proposed methodology to detect the changing profile or absolute values of current might not always indicate a failure of the system. Thus the OCSVM cannot be considered as an automated fault detection algorithm, rather an indicating technique of changing behaviour of the system. The output of OCSVM would probably have to be reviewed to confirm or reject a failure, since at this stage of the research there was a lack of trustworthiness of the data used in the OCSVM training and testing processes.

Secondly, the development of the OCSVM model for an individual point system would have to be carried out when a new point system is considered, since optimal parameters of OCSVM have to be found for each RPS considered.

Finally, the performance of the proposed methodology highly depends on the quality of data used for the training and testing of the OCSVM model to obtain the optimal parameters for the OCSVM to be used as the fault detection model. Thus, one has to consider ways of cleansing the data of discrepancies before using it to train and test the OCSVM model. Otherwise, the obtained results might be very misleading if labelling deficiencies exist in the data.

Three-phase separators

The proposed methodology has to be enhanced to model the separation process by the OOBN, in order to be applicable in practice. In the study performed in this thesis, the separation process was considered to be perfect and instant, i.e. the liquid and gas inflowing in the separator gets separated fully and instantly.

Moreover, the ability to detect leaks in various places of the TPS has to be added to the OOBN. Leaks in different places of the separator might have different effects on the liquid or gas levels in the separator and in some cases might be a critical failure of the separator.

Finally, the testing of the proposed methodology should be done on an actual TPS or a scaled model of TPS. This might indicate some difficulties in the fault detection and diagnostics that cannot be observed using the simulation model. For example, vibration of the equipment could cause random noise in the readings of level or flow transmitters.

5 References

AFONSO, P. A. F. N. A., FERREIRA, J. M. L. & CASTRO, J. A. A. M. 1998. Sensor Fault Detection and Identification in a Pilot Plant Under Process Control. *Chemical Engineering Research and Design,* 76**,** 490-498.

AKBANI, R., KWEK, S. & JAPKOWICZ, N. 2004. Applying Support Vector Machines to Imbalanced Datasets. *In:* BOULICAUT, J.-F., ESPOSITO, F., GIANNOTTI, F. & PEDRESCHI, D. (eds.) *Machine Learning: ECML 2004.* Springer Berlin Heidelberg.

AL-HAJRI, E. M. & ROSSITER, J. A. A unified frame work for oil producing stations using Petri nets. Control 2010, UKACC International Conference on, 7-10 Sept. 2010 2010. 1-8.

AMERICAN PETROLEUM INSTITUTE 2001. API RP 14C: Recommended Practice for Analysis, Design, Installation, and Testing of Basic Surface Safety Systems for Offshore Production Platforms. *Protection concepts and safety analysis.*

ARDAKANI, H. D., LUCAS, C., SIEGEL, D., SHUO, C., DERSIN, P., BONNET, B. & LEE, J. PHM for railway system - A case study on the health assessment of the point machines. Prognostics and Health Management (PHM), 2012 IEEE Conference on, 18-21 June 2012 2012. 1-5.

ARNOLD, K. & STEWART, M. 2008. Chapter 5 - Three-Phase Oil and Water Separation. *In:* STEWART, K. A. (ed.) *Surface Production Operations (Third Edition).* Burlington: Gulf Professional Publishing.

ASADA, T. & ROBERTS, C. Development of an effective condition monitoring system for AC point machines. Railway Condition Monitoring and Non-Destructive Testing (RCM 2011), 5th IET Conference on, 29-30 Nov. 2011 2011. 1-6.

ASADA, T. & ROBERTS, C. 2013. Improving the dependability of DC point machines with a novel condition monitoring system. *Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit*.

ASADA, T., ROBERTS, C. & KOSEKI, T. 2013. An algorithm for improved performance of railway condition monitoring equipment: Alternating-current point machine case study. *Transportation Research Part C: Emerging Technologies,* 30**,** 81-92.

ATAMURADOV, V., CAMCI, F., BASKAN, S. & SEVKLI, M. 2009. Failure Diagnostics for Railway Point Machines Using Expert Systems. *2009 Ieee International Symposium on Diagnostics for Electric Machines, Power Electronics and Drives***,** 335-339.

BAGWELL, P. 2004. The sad state of British railways: The rise and fall of Railtrack, 1992–2002. *Journal of Transport History,* 25**,** 111-124.

BÖHM, T. Accuracy Improvement of Condition Diagnosis of Railway Switches via External Data Integration. *In:* BOLLER, C., ed. Proceedings of the Sixth European Workshop on Structural Health Monitoring, 03.-06. Jul. 2012 2012 Dresden, Deutschland. Deutsche Gesellschaft für Zerstörungsfreie Prüfung, 1550-1558.

BOLBOLAMIRI, N., SANAI, M. S. & MIRABADI, A. 2012. Time-Domain Stator Current Condition Monitoring: Analyzing Point Failures Detection by

Kolmogorov-Smirnov (K-S) Test. *World Academy of Science, Engineering and Technology*, 1049-1055.
BOSER, B. E., GUYON, I. M. & VAPNIK, V. N. 1992. A training algorithm for optimal margin classifiers. *Proceedings of the fifth annual workshop on Computational learning theory.* Pittsburgh, Pennsylvania, USA: ACM.
CDS RAIL. 2013. *Points Condition Monitoring* [Online]. Available: http://www.cdsrail.com/eng/applications/page/Points%20&%20Switch%20Machines.php 2013].
CHAMROUKHI, F., SAMÉ, A. & AKNIN, P. A probabilistic approach for the classification of railway switch operating states. Sixth International Conference on Condition Monitoring and Machinery Failure Prevention Technologies, 2009 Dublin, UK.
CHAMROUKHI, F., SAME, A., AKNIN, P. & ANTONI, M. Switch mechanism diagnosis using a pattern recognition approach. 4th IET International Conference on Railway Condition Monitoring, 18-20 June 2008 2008. 1-4.
CHAMROUKHI, F., SAMÉ, A., GOVAERT, G. & AKNIN, P. 2011. A dynamic probabilistic modeling of railway switches operating states. *Proceedings of the 9th World Congress on Railway Research.* Lille-France.
CHAN, C. W. 2005. An expert decision support system for monitoring and diagnosis of petroleum production and separation processes. *Expert Systems with Applications,* 29, 131-143.
CHANG, C.-C. & LIN, C.-J. 2011. LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.,* 2, 1-27.
CHEN, J. & ROBERTS, C. Effective Condition Monitoring of Line Side Assets. The Institution of Engineering and Technology International Conference on Railway Condition Monitoring, 2006, 29-30 Nov. 2006 2006. 78-83.
CHEN, J., ROBERTS, C. & WESTON, P. 2008. Fault detection and diagnosis for railway track circuits using neuro-fuzzy systems. *Control Engineering Practice,* 16, 585-596.
CHEN, L. & NG, R. 2004. On the marriage of Lp-norms and edit distance. *Proceedings of the Thirtieth international conference on Very large data bases - Volume 30.* Toronto, Canada: VLDB Endowment.
CHEN, L. & ÖZSU, M. T. 2005. Robust and fast similarity search for moving object trajectories. *Proceedings of the 2005 ACM SIGMOD international conference on Management of data.* Baltimore, Maryland: ACM.
DAS, G., GUNOPULOS, D. & MANNILA, H. 1997. Finding similar time series. *Principles of Data Mining and Knowledge Discovery,* 1263, 88-100.
DAUBECHIES, I. 1992. *Ten lectures on wavelets*, Society for Industrial and Applied Mathematics.
DEZA, E. & DEZA, M. 2009. *Encyclopedia of Distances*, Springer.
DIAS, A. C., BHAYA, A. & KASZKUREWICZ, E. Fault Diagnosis in an Oil Production Plant Prototype Using a Diagnostic Model Processor. American Control Conference, 1993, 2-4 June 1993 1993. 107-111.
DONGYU, Z., WANGMENG, Z., ZHANG, D. & HONGZHI, Z. Time Series Classification Using Support Vector Machine with Gaussian Elastic

Metric Kernel. 20th International Conference on Pattern Recognition (ICPR), 23-26 Aug. 2010 2010. 29-32.
EKER, O. F., CAMCI, F., GUCLU, A., YILBOGA, H., SEVKLI, M. & BASKAN, S. 2011. A Simple State-Based Prognostic Model for Railway Turnout Systems. *Ieee Transactions on Industrial Electronics,* 58**,** 1718-1726.
EKER, O. F., CAMCI, F. & KUMAR, U. 2012. SVM Based Diagnostics on Railway Turnouts. *International Journal of Performability Engineering,* 8**,** 289-298.
EMERSON PROCESS MANAGEMENT 2001. *Control valve handbook*, Fisher Controls International LLC.
FIX, E. & HODGES, J. L. 1951. Discriminatory analysis. Nonparametric discrimination: Consistency properties. USAF School of Aviation Medicine, Randolph Field, Texas.
FU, A. W. C., KEOGH, E., LAU, L. Y. H., RATANAMAHATANA, C. A. & WONG, R. C. W. 2008. Scaling and time warping in time series querying. *Vldb Journal,* 17**,** 899-921.
GANYUN, L. V., CHENG, H. Z., ZHAI, H. B. & DONG, L. X. 2005. Fault diagnosis of power transformer based on multi-layer SVM classifier. *Electric Power Systems Research,* 74**,** 1-7.
GAO, Q., HAN, M., HU, S.-L. & DONG, H.-J. Design of Fault Diagnosis System of FPSO Production Process Based on MSPCA. Information Assurance and Security, 2009. IAS '09. Fifth International Conference on, 18-20 Aug. 2009 2009. 729-733.
GUDMUNDSSON, S., RUNARSSON, T. P. & SIGURDSSON, S. Support vector machines and dynamic time warping for time series. Neural Networks, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on, 1-8 June 2008 2008. 2772-2776.
HAAR, A. 1910. Zur Theorie der orthogonalen Funktionensysteme. *Mathematische Annalen,* 69**,** 331-371.
HAYKIN, S. 2008. *Neural Networks and Learning Machines (3rd Edition)*, Prentice Hall.
HOTELLING, H. 1931. The Generalization of Student's Ratio. 360-378.
IMAM, T., TING, K. & KAMRUZZAMAN, J. 2006. z-SVM: An SVM for Improved Classification of Imbalanced Data. *In:* SATTAR, A. & KANG, B.-H. (eds.) *AI 2006: Advances in Artificial Intelligence.* Springer Berlin Heidelberg.
INNOTRACK 2006. List of key parameters for switch and crossing monitoring.
INNOTRACK 2009. Available Sensors for Railway Environments for Condition Monitoring.
JENSEN, F. V. & NIELSEN, T. D. 2007. *Bayesian Networks and Decision Graphs*, Springer.
KALMAN, R. E. 1960. A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME – Journal of Basic Engineering***,** 35-45.
KEOGH, E. & PAZZANI, M. Derivative Dynamic Time Warping. In First SIAM International Conference on Data Mining (SDMâ™2001, 2001.
KEOGH, E., ZHU, Q., HU, B., Y., H., XI, X., WEI, L. & RATANAMAHATANA, C. A. 2011. *The UCR Time Series Classification/Clustering Homepage*

[Online]. Available: http://www.cs.ucr.edu/~eamonn/time_series_data/ 2013].
KINNAERT, M., VRANČIĆ, D., DENOLIN, E., JURIČIĆ, Đ. & PETROVČIĆ, J. 2000. Model-based fault detection and isolation for a gas–liquid separation unit. *Control Engineering Practice,* 8**,** 1273-1283.
KOHAVI, R. 1995. A study of cross-validation and bootstrap for accuracy estimation and model selection. *Proceedings of the 14th international joint conference on Artificial intelligence - Volume 2.* Montreal, Quebec, Canada: Morgan Kaufmann Publishers Inc.
LAMPIS, M. & ANDREWS, J. D. 2009a. Bayesian Belief Networks for System Fault Diagnostics. *Quality and Reliability Engineering International,* 25**,** 409-426.
LAMPIS, M. & ANDREWS, J. D. 2009b. Introducing Dynamics in a Fault Diagnostic Application Using Bayesian Belief Networks. *Proceedings of 2009 8th International Conference on Reliability, Maintainability and Safety, Vols I and Ii***,** 186-190.
LEVENBERG, K. 1944. A method for the solution of certain problems in least squares. *Quart. Applied Math.,* 2**,** 164-168.
LEVENSHTEIN, V. 1966. Binary codes capable of correcting deletions, insertions, and reversals.
LI, Y., FANG, Q., WANG, X., ZHANG, Z. & XIE, C. Research on air-conditioning fault diagnosis method based on SVM. Sixth International Conference on Natural Computation (ICNC), 10-12 Aug. 2010 2010. 3397-3400.
LORENZ, D. & KÖHLER, T. 2005. *A Comparison of Denoising Methods for One Dimensional Time Series*, Zentrum für Technomathematik.
MÁRQUEZ, F. P. G. & CHACÓN MUÑOZ, J. M. 2010. A pattern recognition and data analysis method for maintenance management. *International Journal of Systems Science***,** 1-15.
MÁRQUEZ, F. P. G. & GARCIA-PARDO, I. P. 2010. Principal Component Analysis Applied to Filtered Signals for Maintenance Management. *Quality and Reliability Engineering International,* 26**,** 523-527.
MÁRQUEZ, F. P. G., LEWIS, R. W., TOBIAS, A. M. & ROBERTS, C. 2008. Life cycle costs for railway condition monitoring. *Transportation Research Part E-Logistics and Transportation Review,* 44**,** 1175-1187.
MÁRQUEZ, F. P. G., PAUL, W. & ROBERTS, C. 2007a. Failure analysis and diagnostics for railway trackside equipment. *Engineering Failure Analysis,* 14**,** 1411-1426.
MÁRQUEZ, F. P. G., PEDREGAL, D. J. & ROBERTS, C. 2010. Time series methods applied to failure prediction and detection. *Reliability Engineering & System Safety,* 95**,** 698-703.
MÁRQUEZ, F. P. G. & SCHMID, F. 2007. A digital filter-based approach to the remote condition monitoring of railway turnouts. *Reliability Engineering & System Safety,* 92**,** 830-840.
MÁRQUEZ, F. P. G., SCHMID, F. & COLLADO, J. C. 2003. A reliability centered approach to remote condition monitoring. A railway points case study. *Reliability Engineering & System Safety,* 80**,** 33-40.
MÁRQUEZ, F. P. G., TERCERO, D. J. P. & SCHMID, F. 2007b. Unobserved component models applied to the assessment of wear in railway points:

A case study. *European Journal of Operational Research,* 176**,** 1703-1712.
MARTEAU, P. F. 2009. Time Warp Edit Distance with Stiffness Adjustment for Time Series Matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 31**,** 306-318.
MASSEY, F. J., JR. 1951. The Kolmogorov-Smirnov Test for Goodness of Fit. *Journal of the American Statistical Association,* 46**,** 68-78.
MCCULLOCH, W. & PITTS, W. 1943. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biology,* 5**,** 115-133.
MCHUTCHON, M. A., STASZEWSKI, W. J. & SCHMID, F. 2005. Signal processing for remote condition monitoring of railway points. *Strain,* 41**,** 71-85.
MOSTELLER, F. & TUKEY, J. W. 1968. Data analysis, including statistics. *In:* LINDZEY, G. & ARONSON, E. (eds.) *Handbook of Social Psychology.* 2nd ed. Reading, MA: Addison-Wesley.
NATARAJAN, S. & SRINIVASAN, R. 2010. Multi-model based process condition monitoring of offshore oil and gas production process. *Chemical Engineering Research and Design,* 88**,** 572-591.
NETWORK RAIL 2013. Annual Return 2013. http://www.networkrail.co.uk/publications/Annual-return/.
OMANA, M. & TAYLOR, J. H. Fault Detection and Isolation Using the Generalized Parity Vector Technique in the Absence of an a Priori Mathematical Model. Control Applications, 2007. CCA 2007. IEEE International Conference on, 1-3 Oct. 2007 2007. 970-975.
OYEBANDE, B. O. & RENFREW, A. C. 2002. Condition monitoring of railway electric point machines. *Iee Proceedings-Electric Power Applications,* 149**,** 465-473.
PEARSON, K. 1901. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine,* 2**,** 559-572.
PEDREGAL, D. J., GARCIA, F. P. & ROBERTS, C. 2009. An algorithmic approach for maintenance management based on advanced state space systems and harmonic regressions. *Annals of Operations Research,* 166**,** 109 - 124.
ROVERSO, D. 2002. Plant diagnostics by transient classification: The ALADDIN approach. *International Journal of Intelligent Systems,* 17**,** 767-790.
SAHIN, F., YAVUZ, M. C., ARNAVUT, Z. & ULUYOL, O. 2007. Fault diagnosis for airplane engines using Bayesian networks and distributed particle swarm optimization. *Parallel Computing,* 33**,** 124-143.
SAKOE, H. & CHIBA, S. 1978. Dynamic-Programming Algorithm Optimization for Spoken Word Recognition. *Ieee Transactions on Acoustics Speech and Signal Processing,* 26**,** 43-49.
SCHOLKOPF, B., PLATT, J. C., SHAWE-TAYLOR, J., SMOLA, A. J. & WILLIAMSON, R. C. 2001. Estimating the support of a high-dimensional distribution. *Neural Computation,* 13**,** 1443-1471.
SIEMENS AG. 2011. *Point Diagnostic System SIDIS W* [Online]. Available: http://w1.siemens.ch/home/ch/de/ts/produkteA/produkte/H/Documents/a19100v100b795x7600_sidis_w_e_1366546.pdf 2013].

SILMON, J. A. & ROBERTS, C. 2010. Improving railway switch system reliability with innovative condition monitoring algorithms. *Proceedings of the Institution of Mechanical Engineers Part F-Journal of Rail and Rapid Transit,* 224**,** 293-302.

SKIPSEY, D. 2010. Clamp lock. http://www.rmweb.co.uk/community/index.php?/topic/17767-clamp-locks/.

SMITH, C. L. 2009. *Practical Process Control: Tuning and Troubleshooting*, Wiley.

SPX RAIL SYSTEMS. 2013. *Modular Point Operating Machine SPX Clamplock* [Online]. Available: http://www.spx.com/en/spx-rail-systems/pd-mp-clamplock-point-machine/ 2013].

STRUKTON RAIL. 2013. *POSS online monitoring* [Online]. Available: http://www.struktonrail.com/maintenance/poss-online-monitoring/ 2013].

SVRCEK, W. Y., MAHONEY, D. P. & YOUNG, B. R. 2006. *A Real-Time Approach to Process Control*, John Wiley \\& Sons.

TAYLOR, J. H. & OMANA, M. Fault Detection, Isolation and Accommodation Using the Generalized Parity Vector Technique. *In:* CHUNG, M. J. & MISRA, P., eds. Proceedings of the 17th IFAC World Congress, 2008, 2008 COEX, South Korea 1914-1921.

TAYLOR, J. H. & SAYDA, A. F. Prototype design of a multi-agent system for integrated control and asset management of petroleum production facilities. American Control Conference, 2008, 11-13 June 2008 2008. 4350-4357.

THE OFFICE OF RAIL AND ROAD. 2015. *Network Rail Monitor Quarter 4 of Year 5 of CP4* [Online]. Available: http://orr.gov.uk/what-and-how-we-regulate/regulation-of-network-rail/monitoring-performance/network-rail-monitor [Accessed 2015.07.03.

THINKING FORWARD. 2013. *Integrated Remote System for Monitoring Point Machines* [Online]. Available: http://www.tfxxi.com/index.php/productos/thinking-forward/?lang=en 2013].

TUNLEY, M. 2010. Points to Failure. *IRSE news.* Institution of Railway Signal Engineers.

U.S. ENERGY INFORMATION ADMINISTRATION. 2013. *International Energy Statistics* [Online]. Available: http://www.eia.gov/cfapps/ipdbproject/iedindex3.cfm 2014].

VAPNIK, V. N. 1995. *The nature of statistical learning theory*, Springer-Verlag New York, Inc. .

VILEINISKIS, M., REMENYTE-PRESCOTT, R. & RAMA, D. 2015. A fault detection method for railway point systems. *Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit*.

VILEINISKIS, M., REMENYTE-PRESCOTT, R., RAMA, D. & ANDREWS, J. Fault Diagnostics for Railway Point Machines. *In:* JACKSON, L. & ANDREWS, J., eds. Proceedings of the 20th Advances in Risk and Reliability Technology Symposium, 21-23 May 2013 2013 Loughborough, United Kingdom. 103-119.

VUORI, V., LAAKSONEN, J., OJA, E. & KANGAS, J. Speeding up on-line recognition of handwritten characters by pruning the prototype set.

Document Analysis and Recognition, 2001. Proceedings. Sixth International Conference on, 2001 2001. 501-505.
WXWIDGETS. 2013. *wxWidgets Cross-Platform GUI Library* [Online]. [Accessed 2013.08.01.
XU, B. G. 2012. Intelligent fault inference for rotating flexible rotors using Bayesian belief network. *Expert Systems with Applications,* 39**,** 816-822.
YANG, W. 2006. Sensors and Instrumentation for Monitoring and Control of Multi-Phase Separation. *Measurement and Control,* 39**,** 178-184.
YILBOGA, H., EKER, O. F., GU, X, X00E, LU, A. & CAMCI, F. Failure prediction on railway turnouts using time delay neural networks. IEEE International Conference on Computational Intelligence for Measurement Systems and Applications (CIMSA), 6-8 Sept. 2010 2010. 134-137.
ZATTONI, E. 2006. Detection of incipient failures by using an H-2-norm criterion: Application to railway switching points. *Control Engineering Practice,* 14**,** 885-895.
ZWANENBURG, W. J. Degradation Processes of Switches & Crossings. The Institution of Engineering and Technology International Conference on Railway Condition Monitoring, 29-30 Nov. 2006 2006. 115-119.

6 Appendix A

Table 6.1. Alarm records for PM9B, PM13A and PM13B

PM ID	Total movements	# of movements with alarms	# of motor running duration alarms	# of peak alarms	# of average current alarms
PM9B	575	24	22	0	17
PM13A	519	45	7	0	38
PM13B	521	32	8	0	25

Table 6.2. Failure records for PM9B

Failure occurrence date	Failure rectification date	Failure cause	Maintenance action
05/02/2012 15:05:51	05/02/2012 19:00:01	Obstruction with snow and ice	Ice removed, points lubricated
25/04/2012 08:41:56	25/04/2012 14:38:01	Obstruction with ballast	Ballast removed
09/05/2012 07:49:14	24/05/2012 15:00:01	Obstruction with squashed cans and bottles, bolts missing from slide chairs	Bolts replaced, obstructions removed, points lubricated
14/09/2012 06:52:00	14/09/2012 11:10:01	Contamination	Contamination cleared, points lubricated
22/11/2012 10:22:11	24/11/2012 04:48:01	Crushed tail cable	Cable replaced

Table 6.3. Failure records for PM13A

Failure occurrence date	Failure rectification date	Failure cause	Maintenance action
07/02/2012 20:59:45	08/02/2012 14:00:58	Dry slide chairs and crushed ballast	Points cleaned and slide chairs lubricated
04/12/2012 10:22:41	11/01/2013 17:13:56	Alarm system thresholds needs seasonal adjustment	Alarm system thresholds adjusted

Table 6.4. Failure records for PM13B

Failure occurrence date	Failure rectification date	Failure cause	Maintenance action
07/02/2012 20:59:45	08/02/2012 14:00:58	Dry slide chairs and crushed ballast	Points cleaned and slide chairs lubricated
04/12/2012 10:22:41	11/01/2013 17:13:56	Alarm system thresholds needs seasonal adjustment	Alarm system thresholds adjusted

7 Appendix B

Results with time series similarity measure as a kernel distance function

Table 7.1. Classification rates with Euclidean distance for PM9B (5-fold cross validation, labelling using alarm dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	7	25	0	0
$90 \leq P_F < 95$	27	0	0	0
$85 \leq P_F < 90$	0	1	0	0
$P_F < 85$	0	0	0	0

Table 7.2. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9B (5-fold cross validation, labelling using alarm dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	29	0	0	$P_F \geq 95$	1	30	0	0
$90 \leq P_F < 95$	2	2	1	0	$90 \leq P_F < 95$	0	0	1	9
$85 \leq P_F < 90$	0	4	6	2	$85 \leq P_F < 90$	0	3	4	1
$P_F < 85$	0	6	7	1	$P_F < 85$	0	0	11	0

Table 7.3. Classification rates with Euclidean distance for PM13A (5-fold cross validation, labelling using alarm dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	3	57

Table 7.4. Classification rates with ERP distance (on non-scaled and rescaled data) for PM13A (5-fold cross validation, labelling using alarm

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	2	$85 \leq P_F < 90$	0	0	0	1
$P_F < 85$	0	0	0	58	$P_F < 85$	0	1	6	52

Table 7.5. Classification rates with Euclidean distance for PM13B (5-fold cross validation, labelling using alarm dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	4	3
$P_F < 85$	0	2	14	37

Table 7.6. Classification rates with ERP distance (on non-scaled and rescaled data) for PM13B (5-fold cross validation, labelling using alarm dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	8	0	$90 \leq P_F < 95$	0	6	1	0
$85 \leq P_F < 90$	0	2	18	0	$85 \leq P_F < 90$	0	21	1	1
$P_F < 85$	0	1	1	30	$P_F < 85$	0	1	2	27

Table 7.7. Classification rates with Euclidean distance for PM9B (5-fold cross validation, labelling using failure dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60

Table 7.8. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9B (5-fold cross validation, labelling using failure dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	60

Table 7.9. Classification rates with Euclidean distance for PM13A (5-fold cross validation, labelling using failure dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	36
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	1	0	23

Table 7.10. Classification rates with ERP distance (on non-scaled and rescaled data) for PM13A (5-fold cross validation, labelling using failure dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	6
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	54

Table 7.11. Classification rates with Euclidean distance for PM13B (5-fold cross validation, labelling using failure dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60

Table 7.12. Classification rates with ERP distance (on non-scaled and rescaled data) for PM13B (5-fold cross validation, labelling using failure dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	1	59

Table 7.13. Classification rates with Euclidean distance for PM9B (80% training and 20% testing, labelling using alarm dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	50	6	0	0
$90 \leq P_F < 95$	4	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	0

Table 7.14. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9B (80% training and 20% testing, labelling using alarm dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	30	6	1	0	$P_F \geq 95$	30	9	1	0
$90 \leq P_F < 95$	2	12	1	0	$90 \leq P_F < 95$	5	15	0	0
$85 \leq P_F < 90$	0	8	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	0	$P_F < 85$	0	0	0	0

Table 7.15. Classification rates with Euclidean distance for PM13A (80% training and 20% testing, labelling using alarm dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	6
$P_F < 85$	0	0	8	46

Table 7.16. Classification rates with ERP distance (on non-scaled and rescaled data) for PM13A (80% training and 20% testing, labelling using alarm dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	3	$90 \leq P_F < 95$	0	0	0	1
$85 \leq P_F < 90$	0	0	0	5	$85 \leq P_F < 90$	0	0	0	4
$P_F < 85$	0	0	5	47	$P_F < 85$	0	0	4	51

Table 7.17. Classification rates with Euclidean distance for PM13B (80% training and 20% testing, labelling using alarm dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	2
$85 \leq P_F < 90$	0	0	0	2
$P_F < 85$	0	2	6	48

Table 7.18. Classification rates with ERP distance (on non-scaled and rescaled data) for PM13B (80% training and 20% testing, labelling using alarm dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	2	18	0	0	$P_F \geq 95$	0	1	3	4
$90 \leq P_F < 95$	6	4	0	0	$90 \leq P_F < 95$	0	2	7	2
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	3	7	0
$P_F < 85$	0	0	6	24	$P_F < 85$	0	0	0	31

Table 7.19. Classification rates with Euclidean distance for PM9A (60% training and 40% testing, labelling using alarm dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	3	9	1	30
$90 \leq P_F < 95$	2	7	1	0
$85 \leq P_F < 90$	3	4	0	0
$P_F < 85$	0	0	0	0

Table 7.20. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9A (60% training and 40% testing, labelling using alarm dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	7	1	0	$P_F \geq 95$	0	1	1	3
$90 \leq P_F < 95$	21	25	0	0	$90 \leq P_F < 95$	1	7	4	0
$85 \leq P_F < 90$	2	3	0	0	$85 \leq P_F < 90$	1	9	5	0
$P_F < 85$	1	0	0	0	$P_F < 85$	2	6	15	5

Table 7.21. Classification rates with Euclidean distance for PM9B (60% training and 40% testing, labelling using alarm dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	47	9	0	0
$90 \leq P_F < 95$	4	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	0

Table 7.22. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9B (60% training and 40% testing, labelling using alarm dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	29	1	7	0	$P_F \geq 95$	23	0	4	0
$90 \leq P_F < 95$	2	1	1	0	$90 \leq P_F < 95$	9	6	10	0
$85 \leq P_F < 90$	0	5	10	1	$85 \leq P_F < 90$	0	3	4	0
$P_F < 85$	0	2	1	0	$P_F < 85$	0	0	1	0

Table 7.23. Classification rates with Euclidean distance for PM13A (60% training and 40% testing, labelling using alarm dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	3	57

Table 7.24. Classification rates with ERP distance (on non-scaled and rescaled data) for PM13A (60% training and 40% testing, labelling using alarm dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	60

Table 7.25. Classification rates with Euclidean distance for PM13B (60% training and 40% testing, labelling using alarm dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	1	2	0
$85 \leq P_F < 90$	0	3	4	1
$P_F < 85$	0	0	6	43

Table 7.26. Classification rates with ERP distance (on non-scaled and rescaled data) for PM13B (60% training and 40% testing, labelling using alarm dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	2	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	3	1	$90 \leq P_F < 95$	0	3	3	4
$85 \leq P_F < 90$	0	0	4	1	$85 \leq P_F < 90$	0	7	5	2
$P_F < 85$	0	3	15	31	$P_F < 85$	0	1	5	30

Table 7.27. Classification rates with Euclidean distance for PM9A (40% training and 60% testing, labelling using alarm dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	5	0	9
$90 \leq P_F < 95$	3	11	0	0
$85 \leq P_F < 90$	3	5	1	19
$P_F < 85$	1	0	0	3

Table 7.28. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9A (40% training and 60% testing, labelling using alarm dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	9	0	0	$P_F \geq 95$	0	0	0	2
$90 \leq P_F < 95$	19	7	1	0	$90 \leq P_F < 95$	1	5	0	2
$85 \leq P_F < 90$	8	12	1	0	$85 \leq P_F < 90$	1	7	1	9
$P_F < 85$	2	0	0	1	$P_F < 85$	3	1	9	19

Table 7.29. Classification rates with Euclidean distance for PM9B (40% training and 60% testing, labelling using alarm dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	5	23	15	0
$90 \leq P_F < 95$	8	1	0	0
$85 \leq P_F < 90$	8	0	0	0
$P_F < 85$	0	0	0	0

Table 7.30. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9B (40% training and 60% testing, labelling using alarm dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	21	11	2	1	$P_F \geq 95$	1	8	1	4
$90 \leq P_F < 95$	1	3	0	3	$90 \leq P_F < 95$	2	11	6	0
$85 \leq P_F < 90$	0	0	3	1	$85 \leq P_F < 90$	0	2	7	1
$P_F < 85$	9	0	3	2	$P_F < 85$	6	8	2	1

Table 7.31. Classification rates with Euclidean distance for PM13A (40% training and 60% testing, labelling using alarm dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	2	58

Table 7.32. Classification rates with ERP distance (on non-scaled and rescaled data) for PM13A (40% training and 60% testing, labelling using alarm dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	1	1	1	57	$P_F < 85$	0	2	0	58

Table 7.33. Classification rates with Euclidean distance for PM13B (40% training and 60% testing, labelling using alarm dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	1	1
$85 \leq P_F < 90$	0	1	4	1
$P_F < 85$	0	0	3	49

Table 7.34. Classification rates with ERP distance (on non-scaled and rescaled data) for PM13B (40% training and 60% testing, labelling using alarm dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	1
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	1
$P_F < 85$	0	0	3	57	$P_F < 85$	0	0	0	58

Table 7.35. Classification rates with Euclidean distance for PM9A (20% training and 80% testing, labelling using alarm dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	1	44
$90 \leq P_F < 95$	0	1	1	5
$85 \leq P_F < 90$	1	3	1	0
$P_F < 85$	3	0	0	0

Table 7.36. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9A (20% training and 80% testing, labelling using alarm dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	1	0	45	$P_F \geq 95$	0	0	0	26
$90 \leq P_F < 95$	0	0	0	9	$90 \leq P_F < 95$	1	0	0	28
$85 \leq P_F < 90$	0	0	0	1	$85 \leq P_F < 90$	0	0	0	3
$P_F < 85$	2	1	0	1	$P_F < 85$	0	0	0	2

Table 7.37. Classification rates with Euclidean distance for PM9B (20% training and 80% testing, labelling using alarm dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	15	13	1	30
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	1	0	0
$P_F < 85$	0	0	0	0

Table 7.38. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9B (20% training and 80% testing, labelling using alarm dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	4	44	$P_F \geq 95$	0	0	0	38
$90 \leq P_F < 95$	0	0	0	4	$90 \leq P_F < 95$	0	0	0	5
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	2
$P_F < 85$	2	1	1	4	$P_F < 85$	3	4	0	8

Table 7.39. Classification rates with Euclidean distance for PM13A (20% training and 80% testing, labelling using alarm dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	2	5	53

Table 7.40. Classification rates with ERP distance (on non-scaled and rescaled data) for PM13A (20% training and 80% testing, labelling using alarm dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	5	55	$P_F < 85$	0	0	6	54

Table 7.41. Classification rates with Euclidean distance for PM13B (20% training and 80% testing labelling using alarm dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	4	56

Table 7.42. Classification rates with ERP distance (on non-scaled and rescaled data) for PM13B (20% training and 80% testing, labelling using alarm dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	7	1	52	$P_F < 85$	0	0	0	60

Table 7.43. Classification rates with Euclidean distance for PM9B (80% training and 20% testing, labelling using failure dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	1
$P_F < 85$	0	0	0	59

Table 7.44. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9B (80% training and 20% testing, labelling using failure dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	2	17	14	27	$P_F < 85$	2	6	6	46

Table 7.45. Classification rates with Euclidean distance for PM13A (80% training and 20% testing, labelling using failure dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	29	31

Table 7.46. Classification rates with ERP distance (on non-scaled and rescaled data) for PM13A (80% training and 20% testing, labelling using failure dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	18	42	$P_F < 85$	0	0	7	53

Table 7.47. Classification rates with Euclidean distance for PM13B (80% training and 20% testing, labelling using failure dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60

Table 7.48. Classification rates with ERP distance (on non-scaled and rescaled data) for PM13B (80% training and 20% testing, labelling using failure dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	60

Table 7.49. Classification rates with Euclidean distance for PM9A (60% training and 40% testing, labelling using failure dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60

Table 7.50. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9A (60% training and 40% testing, labelling using failure dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	2
$P_F < 85$	0	0	1	59	$P_F < 85$	0	0	0	58

Table 7.51. Classification rates with Euclidean distance for PM9B (60% training and 40% testing, labelling using failure dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60

Table 7.52. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9B (60% training and 40% testing, labelling using failure dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	3	57	$P_F < 85$	0	2	3	55

Table 7.53. Classification rates with Euclidean distance for PM13A (60% training and 40% testing, labelling using failure dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	17	43

Table 7.54. Classification rates with ERP distance (on non-scaled and rescaled data) for PM13A (60% training and 40% testing, labelling using failure dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	2
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	3	57	$P_F < 85$	0	0	0	58

Table 7.55. Classification rates with Euclidean distance for PM13B (60% training and 40% testing, labelling using failure dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	1
$90 \leq P_F < 95$	0	0	0	3
$85 \leq P_F < 90$	0	0	0	5
$P_F < 85$	0	0	15	36

Table 7.56. Classification rates with ERP distance (on non-scaled and rescaled data) for PM13B (60% training and 40% testing, labelling using failure dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	4	1	55

Table 7.57. Classification rates with Euclidean distance for PM9A (40% training and 60% testing, labelling using failure dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60

Table 7.58. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9A (40% training and 60% testing, labelling using failure dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	60

Table 7.59. Classification rates with Euclidean distance for PM9B (40% training and 60% testing, labelling using failure dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60

Table 7.60. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9B (40% training and 60% testing, labelling using failure dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	60

Table 7.61. Classification rates with Euclidean distance for PM13A (40% training and 60% testing, labelling using failure dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60

Table 7.62. Classification rates with ERP distance (on non-scaled and rescaled data) for PM13A (40% training and 60% testing, labelling using failure dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	60

Table 7.63. Classification rates with Euclidean distance for PM13B (40% training and 60% testing, labelling using failure dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	2
$90 \leq P_F < 95$	0	0	0	1
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	57

Table 7.64. Classification rates with ERP distance (on non-scaled and rescaled data) for PM13B (40% training and 60% testing, labelling using failure dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	60

Table 7.65. Classification rates with Euclidean distance for PM9A (20% training and 80% testing, labelling using failure dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60

Table 7.66. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9A (20% training and 80% testing, labelling using failure dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	60

Table 7.67. Classification rates with Euclidean distance for PM9B (20% training and 80% testing, labelling using failure dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60

Table 7.68. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9B (20% training and 80% testing, labelling using failure dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	60

Table 7.69. Classification rates with Euclidean distance for PM13A (20% training and 80% testing, labelling using failure dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60

Table 7.70. Classification rates with ERP distance (on non-scaled and rescaled data) for PM13A (20% training and 80% testing, labelling using failure dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	4	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	3	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	53	$P_F < 85$	0	0	0	60

Table 7.71. Classification rates with Euclidean distance for PM13B (20% training and 80% testing labelling using failure dates)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	10
$85 \leq P_F < 90$	0	0	0	4
$P_F < 85$	0	0	0	46

Table 7.72. Classification rates with ERP distance (on non-scaled and rescaled data) for PM13B (20% training and 80% testing, labelling using failure dates)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	1	59	$P_F < 85$	0	0	0	60

8 Appendix C

Analysis of discrepancies in the data labelling based on alarm dates

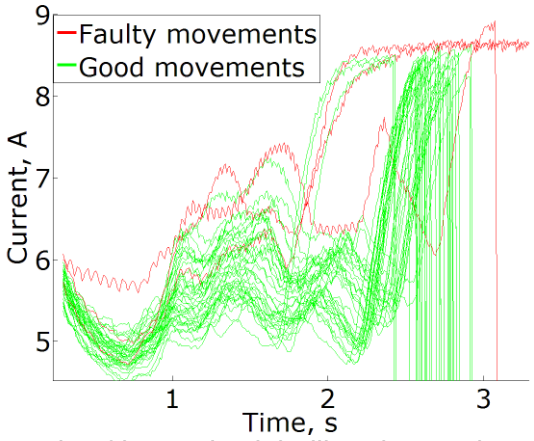


Figure 8.1. Example of imprecise labelling due to alarm dates on PM9A

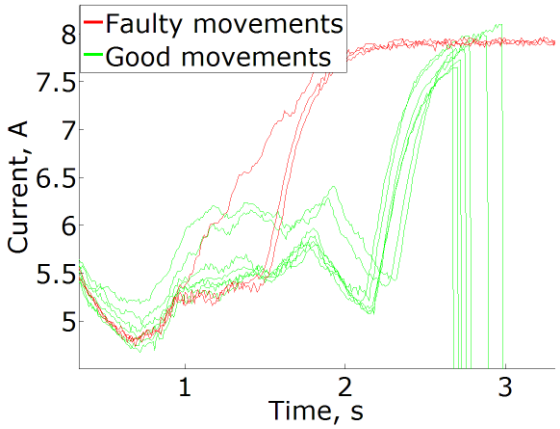


Figure 8.2. Example of imprecise labelling due to alarm dates on PM9B

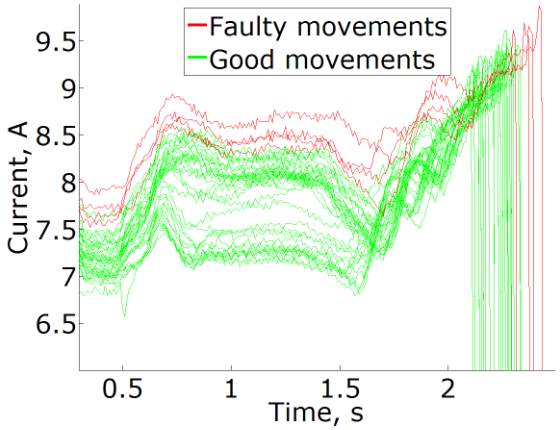


Figure 8.3. Example of imprecise labelling due to alarm dates on PM13A

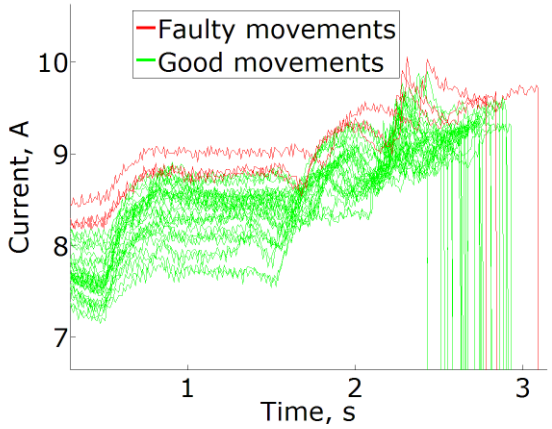


Figure 8.4. Example of imprecise labelling due to alarm dates on PM13B

Analysis of discrepancies in the data labelling based on failure dates

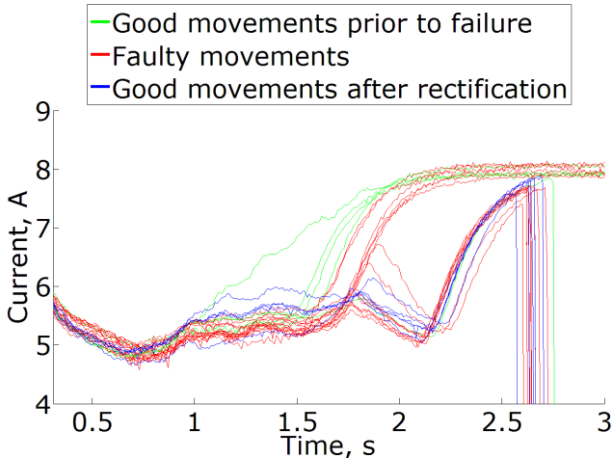


Figure 8.5. Good movements and faulty movements with the obstruction of snow

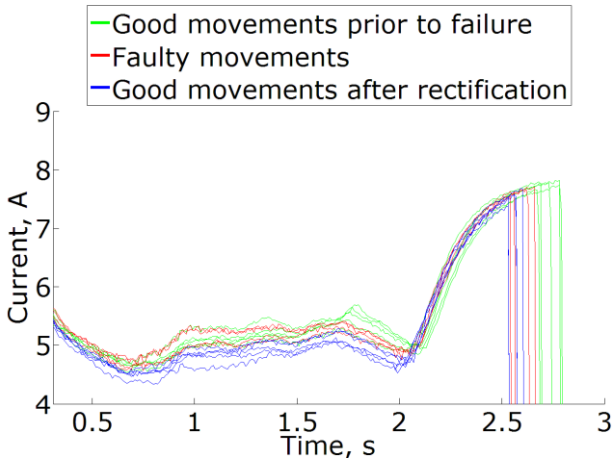


Figure 8.6. Good movements and faulty movements with the obstruction of ballast

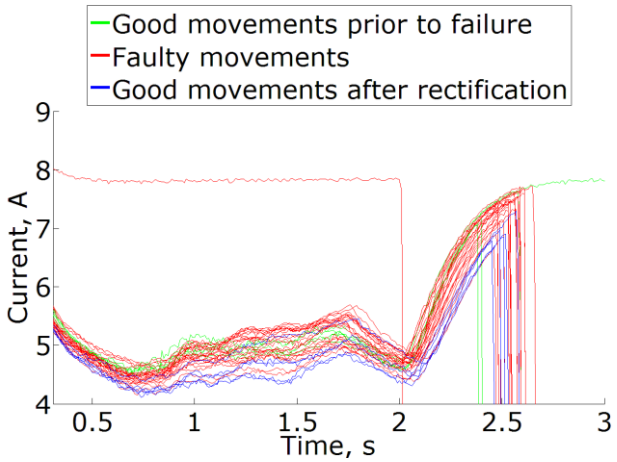


Figure 8.7. Good movements and faulty movements with broken bolt in the crossing block together with obstruction

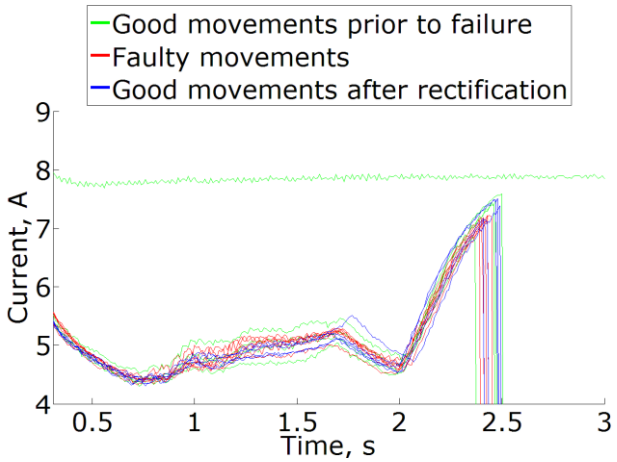


Figure 8.8. Good movements and faulty movements with contamination

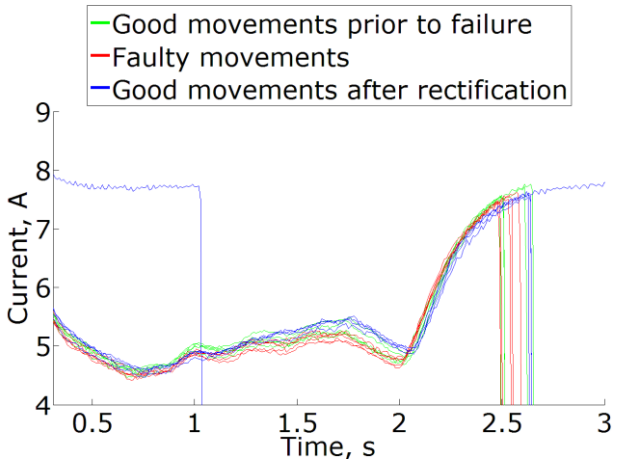


Figure 8.9. Good movements and faulty movements with contamination

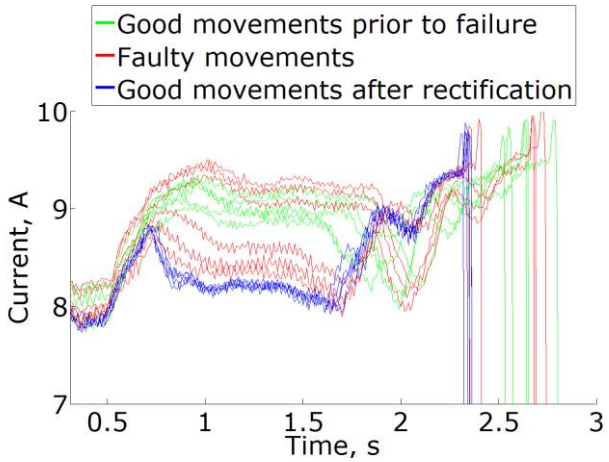


Figure 8.10. Good movements and faulty movements with dry slide chairs and crushed ballast

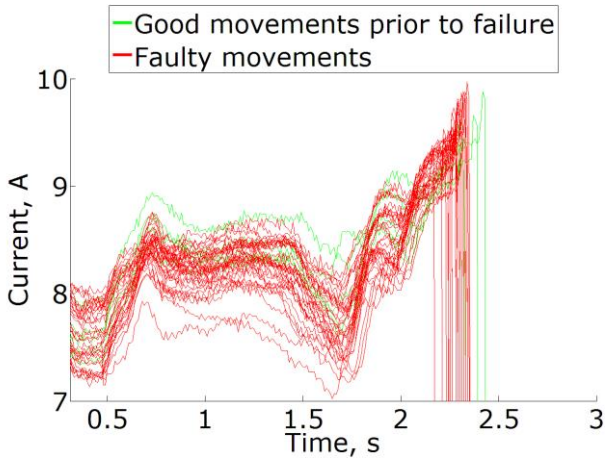


Figure 8.11. Good movements and faulty movements which needed seasonal adjustments for the threshold values

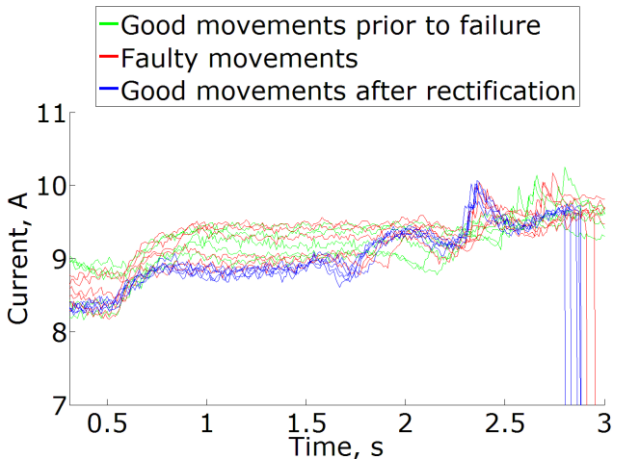


Figure 8.12. Good movements and faulty movements with dry slide chairs and crushed ballast

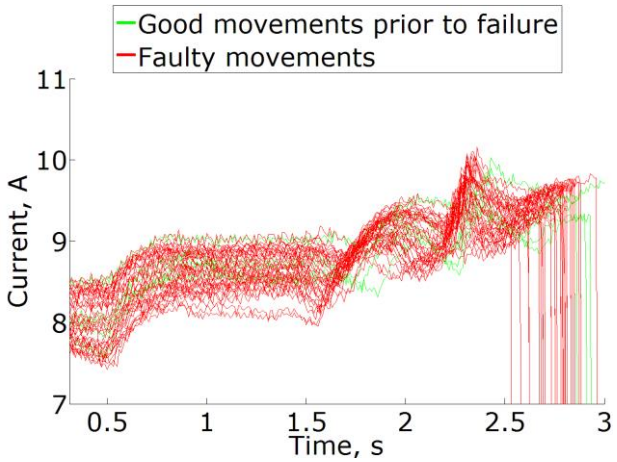


Figure 8.13. Good movements and faulty movements which needed seasonal adjustments for the threshold values

9 Appendix D

Results with time series similarity measure as a feature input (with 1 previous movement) to OCSVM

Table 9.1. Classification rates with Euclidean distance for PM9A (5-fold cross validation, relabelled data)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60

Table 9.2. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9A (5-fold cross validation, relabelled data)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	60

Table 9.3. Classification rates with ERP distance combined with Euclidean distance (on non-scaled and rescaled data) for PM9A (5-fold cross validation, relabelled data)

δ_{EE}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{EE}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	60

Table 9.4. Classification rates with DTW distance (on non-scaled and rescaled data) for PM9A (5-fold cross validation, relabelled data)

δ_{DTW}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{DTW}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	60

Table 9.5. Classification rates with DTW distance combined with Euclidean distance (on non-scaled and rescaled data) for PM9A (5-fold cross validation, relabelled data)

δ_{DE}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{DE}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	60

Results with time series similarity measure as a feature input (with 2 previous movements) to OCSVM

Table 9.6. Classification rates with Euclidean distance for PM9A (5-fold cross validation, relabelled data)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60

Table 9.7. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9A (5-fold cross validation, relabelled data)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	60

Table 9.8. Classification rates with ERP distance combined with Euclidean distance (on non-scaled and rescaled data) for PM9A (5-fold cross validation, relabelled data)

δ_{EE}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{EE}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	60

Table 9.9. Classification rates with DTW distance (on non-scaled and rescaled data) for PM9A (5-fold cross validation, relabelled data)

δ_{DTW}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{DTW}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	60

Table 9.10. Classification rates with DTW distance combined with Euclidean distance (on non-scaled and rescaled data) for PM9A (5-fold cross validation, relabelled data)

δ_{DE}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{DE}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	60

Results with time series similarity measure as a feature input (with 10 previous movements) to OCSVM

Table 9.11. Classification rates with Euclidean distance for PM9A (5-fold cross validation, relabelled data)

δ_E^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60

Table 9.12. Classification rates with ERP distance (on non-scaled and rescaled data) for PM9A (5-fold cross validation, relabelled data)

δ_{ERP}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{ERP}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	60

Table 9.13. Classification rates with ERP distance combined with Euclidean distance (on non-scaled and rescaled data) for PM9A (5-fold cross validation, relabelled data)

δ_{EE}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{EE}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	.	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	60

Table 9.14. Classification rates with DTW distance (on non-scaled and rescaled data) for PM9A (5-fold cross validation, relabelled data)

δ_{DTW}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{DTW}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	0
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	60

Table 9.15. Classification rates with DTW distance combined with Euclidean distance (on non-scaled and rescaled data) for PM9A (5-fold cross validation, relabelled data)

δ_{DE}	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$	δ_{DE}^R	$P_G \geq 95$	$90 \leq P_G < 95$	$85 \leq P_G < 90$	$P_G < 85$
$P_F \geq 95$	0	0	0	0	$P_F \geq 95$	0	0	0	0
$90 \leq P_F < 95$	0	0	0	0	$90 \leq P_F < 95$	0	0	0	0
$85 \leq P_F < 90$	0	0	0	0	$85 \leq P_F < 90$	0	0	0	2
$P_F < 85$	0	0	0	60	$P_F < 85$	0	0	0	58

10 Appendix E

Table 10.1. Summary of BBN for the flow through valve indicated by flow rate transmitter FT given level command indicated by level transmitter LT

Node label	Description	Node states
"LT info"	Represents the level information transformed into a PI controller output command. A prior probability table has to be specified for this node. **Parent of nodes**: *"LT sends info to LC"* **Child of nodes**: none **Type of node:** Information node **Input/output of the BBN:** Input	1 - CV 0, 2 - CV (0-20], 3 - CV (20-40], 4 - CV (40-60), 5 - CV [60-80), 6 - CV [80-100), 7 - CV 100
"LT failure"	Represents the condition of the transmitter LT. The first state of the node represents the non-failed condition, while the remaining states can be used to represent all the possible failure modes (in this case only one failure mode – failed stuck). The prior probabilities for each state have to be specified. **Parent of nodes**: *"LT sends info to LC"* **Child of nodes**: none **Type of node:** Condition node **Input/output of the BBN:** Input	1 – LT W, 2 – LT FS
"LT sends info to LC"	Node to model the information sent from transmitter LT to a controller LC. The probability table of this node shows the probability for a transmitter LT to send specific information to LC, given the condition of the transmitter *"LT failure"* and the information of transmitter LT *"LT info"*. For example, if the node *"LT failure"* is in a state "1 – LT W" and the node *"LT info"* is in a state *"4 - CV (40-60)"*, the state of node *"LT sends info to LC"* will be *"4 - CV (40-60)".* **Parent of nodes**: *"LC sends command to CV"* **Child of nodes**: *"LT info", "LT failure"* **Type of node:** none **Input/output of the BBN:** none	1 - CV 0, 2 - CV (0-20], 3 - CV (20-40], 4 - CV (40-60), 5 - CV [60-80), 6 - CV [80-100), 7 - CV 100
"LC failure"	Represents the condition of the controller LC. First state of the node represents the non-failed condition, while the remaining states can be used to represent all the possible failure modes (in this case two failure modes – failed low and failed high). The prior probabilities for each state have to be specified. **Parent of nodes**: *"LC sends command to CV"* **Child of nodes**: none	1 – LC W, 2 – LC FL, 3 – LC FH

Node label	Description	Node states
	Type of node: Condition node **Input/output of the BBN:** Input	
"LC sends command to CV"	Node to model the command sent to the valve CV from the controller LC. The probability table of this node shows the probability for a controller LC to send a specific command to CV, given the condition of the controller *"LC failure"* and the level received from transmitter LT *"LT sends info to LC"*. For example, if the node *"LC failure"* is in a state *"1 – LC W"* and the node *"LT sends info to LC"* is in a state *"4 - CV (40-60)"*, the state of the node *"LC sends command to CV"* will be *"4 - CV (40-60)"*. **Parent of nodes**: *"CV opening"* **Child of nodes**: *"LT sends info to LC", "LC failure"* **Type of node:** none **Input/output of the BBN:** none	1 - CV 0, 2 - CV (0-20], 3 - CV (20-40], 4 - CV (40-60), 5 - CV [60-80), 6 - CV [80-100), 7 - CV 100
"CV failure"	Represents the condition of the valve CV. First state of the node represents the non-failed condition, while the remaining states can be used to represent all the possible failure modes (in this case two failure modes – failed closed and failed opened). The prior probabilities for each state have to be specified. **Parent of nodes**: *"CV opening"* **Child of nodes**: none **Type of node:** Condition node **Input/output of the BBN:** Input	1 – CV W, 2 – CV FC, 3 – CV FO
"CV opening"	Node to model the opening of the valve CV. The probability table of this node shows the probability for a valve CV to be in a specific opening, given the condition of the valve *"CV failure"* and the command sent to the valve *"LC sends command to CV"*. For example, if the node *"CV failure"* is in a state *"3 – CV FO"*, the state of the node *"CV opening"* will be *"7 – CV 100"* no matter what is the state of the node *"LC sends command to CV"*. **Parent of nodes**: *"FT info"* **Child of nodes**: *"LC sends command to CV", "CV failure"* **Type of node:** none **Input/output of the BBN:** Output	1 - CV 0, 2 - CV (0-20], 3 - CV (20-40], 4 - CV (40-60), 5 - CV [60-80), 6 - CV [80-100), 7 - CV 100
"FT failure"	Represents the condition of the flow transmitter FT. First state of the node represents the non-failed condition, while the remaining states can be used to represent all the possible failure modes (in this case one failure mode – failed stuck). The prior probabilities for each state have to be specified. **Parent of nodes**: *"FT info"* **Child of nodes**: none **Type of node:** Condition node **Input/output of the BBN:** Input	1 – FT W, 2 – FT FS

Node label	Description	Node states
"FT info"	Node to model the flow information shown by the flow transmitter FT. The probability table of this node shows the probability for a flow transmitter to show certain information given the condition of the flow transmitter "FT failure" and the opening of valve *"CV opening"*. For example, if the node *"FT failure"* is in the state *"2 – FT FS"*, then the node *"FT info"* can be in any state, regardless of the state of the node *"CV opening"*. **Parent of nodes**: none **Child of nodes**: *"CV opening", "FT failure"* **Type of node:** Information node **Input/output of the BBN:** Output	1 - CV 0, 2 - CV (0-20], 3 - CV (20-40], 4 - CV (40-60), 5 - CV [60-80), 6 - CV [80-100), 7 - CV 100

Table 10.2. Summary of the BBN *"Separation section"*

Node label	Description	Node states
"Water inflow"	Rule 1 was used to create this node. It represents the inflowing water into the separation section. If certain knowledge is available about the inflowing water, this can be represented by increasing the probabilities of specific states of the node. It is assumed that different inflows have equal probabilities, thus each state of the node has the same prior probability. **Parent of nodes**: *"FT0 water inflow info"*, *"Water level change rate", "FT0 water – FT1", "FT0 water – FT1 + FT0 oil", "Flow over weir", "Total liquid level change rate"* **Child of nodes**: none **Type of node:** none **Input/output of the BBN:** Output	1: CV 0, 2: CV (0-20], 3: CV (20-40], 4: CV (40-60), 5: CV [60-80), 6: CV [80-100), 7: CV 100
"Oil inflow"	Rule 1 was used to create this node. It represents the inflowing oil into the separation section. If certain knowledge is available about the inflowing oil, this can be represented by increasing the probabilities of specific states of the node. It is assumed that different inflows have equal probabilities, thus each state of the node has the same prior probability. **Parent of nodes**: *"FT0 oil inflow info", "FT0 water – FT1 + FT0 oil", "Flow over weir", "Total liquid level change rate"* **Child of nodes**: none **Type of node:** none **Input/output of the BBN:** Output	1: CV 0, 2: CV (0-20], 3: CV (20-40], 4: CV (40-60), 5: CV [60-80), 6: CV [80-100), 7: CV 100
"FT0 failure"	Rule 2 was used to create this node. Note that the flow transmitter FT0 is a multi-flow transmitter, thus only one failure node is created. It represents the condition of flow transmitter FT0. First state	1: FT W, 2: FT FS

Node label	Description	Node states
	of the node represents the non-failed condition (FT W), while the second state represents the failure mode – failed stuck (FT FS). This node is linked to any nodes, which represent the information shown by the flow transmitter FT0, so that the failure effects of specific failure modes could be modelled. **Parent of nodes**: *"FT0 water inflow info", "FT0 oil inflow info", "FT0 water – FT1", "FT0 water – FT1 + FT0 oil"* **Child of nodes**: none **Type of node:** Condition node **Input/output of the BBN:** Input	
"FT0 water inflow info"	Rule 3 was used to create this node. It is used to input the readings from the flow transmitter FT0 about the water flow into the separator. The probability table of this node is used to model the failure effects of flow transmitter FT0 on the information that it is showing about the water flow. **Parent of nodes**: none **Child of nodes**: *"FT0 failure", "Water inflow"* **Type of node:** Information node **Input/output of the BBN:** Output	1: CV 0, 2: CV (0-20], 3: CV (20-40], 4: CV (40-60), 5: CV [60-80), 6: CV [80-100), 7: CV 100
"FT0 oil inflow info"	Rule 3 was used to create this node. It is used to input the readings from the flow transmitter FT0 about the oil flow into the separator. The probability table of this node is used to model the failure effects of flow transmitter FT0 on the information that it is showing about the oil flow. **Parent of nodes**: none **Child of nodes**: *"FT0 failure", "Oil inflow"* **Type of node:** Information node **Input/output of the BBN:** Output	1: CV 0, 2: CV (0-20], 3: CV (20-40], 4: CV (40-60), 5: CV [60-80), 6: CV [80-100), 7: CV 100
"FT1 given level LT1"	Rule 4 was used to create this node. This is an instance node to model the interface level control loop. **Inputs of node**: *"LT info", "LT failure", "LC failure", "CV failure", "FT failure"* **Outputs of node**: *"FT info", "CV opening"* **Type of node:** Instance node **Input/output of the BBN:** none	-
"LT1 info"	Rule 4.a was used to create this node. It represents the water level information transformed into a PI controller output command. **Parent of nodes**: *"FT1 given level LT1.LT info"* **Child of nodes**: none **Type of node:** Information node	1: CV 0, 2: CV (0-20], 3: CV (20-40], 4: CV (40-60), 5: CV [60-80),

Node label	Description	Node states
	Input/output of the BBN: Input	6: CV [80-100), 7: CV 100
"LT1 failure"	Rule 4.b was used to create this node. It represents the condition of the transmitter LT1. The first state of the node represents the non-failed condition (LT W), while the second state represents the failure mode – failed stuck (LT FS). This node is linked to any nodes, which represent the information shown by the LT1 transmitter, so that the failure effects of specific failure modes could be modelled. **Parent of nodes**: *"FT1 given level LT1.LT failure", "LT1 water level change rate", "LT1 total liquid level", "LT1 total liquid level change rate"* **Child of nodes**: none **Type of node:** Condition node **Input/output of the BBN:** Input	1: LT W, 2: LT FS,
"LC1 failure"	Rule 4.c was used to create this node. It represents the condition of the controller LC1. First state of the node represents the non-failed condition (LC W), while the remaining states represent the two failure modes – failed low (LC FL) and failed high (LC FH). This node is linked to any nodes, which represent the command that the controller LC1 sends to the valve CV1, so that the failure effects of specific failure modes could be modelled. **Parent of nodes**: *"FT1 given level LT1.LC failure"* **Child of nodes**: none **Type of node:** Condition node **Input/output of the BBN:** Input	1: LC W, 2: LC FL, 3: LC FH
"CV1 failure"	Rule 4.d was used to create this node. It represents the condition of the valve CV1. First state of the node represents the non-failed condition (CV W), while the remaining states represent the two failure modes – failed closed (CV FC) and failed opened (CV FO). This node is linked to any nodes, which represent the opening of valve CV1, so that the failure effects of specific failure modes could be modelled. **Parent of nodes**: *"FT1 given level LT1.CV failure"* **Child of nodes**: none **Type of node:** Condition node **Input/output of the BBN:** Input	1: CV W, 2: CV FC, 3: CV FO
"FT1 failure"	Rule 4.e was used to create this node. It represents the condition of the flow transmitter FT1. First state of the node represents the non-failed condition (FT W), while the second state represents the failure mode – failed stuck (FT FS) This node is linked to any nodes, which represent the information shown by the flow transmitter FT1, so that the failure effects of specific failure modes	1: FT W, 2: FT FS

Node label	Description	Node states
	could be modelled. **Parent of nodes**: *"FT1 given level LT1.FT failure", "FT0 water – FT1", "FT0 water – FT1 + FT0 oil"* **Child of nodes**: none **Type of node:** Condition node **Input/output of the BBN:** Input	
"FT1 info"	Rule 4.f was used to create this node. It is used to input the readings from the flow transmitter FT1 about the water outflow from the separator. **Parent of nodes**: none **Child of nodes**: *"FT1 given level LT1.FT info"* **Type of node:** Information node **Input/output of the BBN:** Output	1: CV 0, 2: CV (0-20], 3: CV (20-40], 4: CV (40-60), 5: CV [60-80), 6: CV [80-100), 7: CV 100
"Water outflow"	Rule 4.g was used to create this node. It is used to obtain the information from the output node *"CV opening"* of the instance node *"FT1 given level LT1"*. It is further used to model the water level difference in the separation section. **Parent of nodes**: *"Water level change rate", "FT0 water – FT1", "FT0 water – FT1 + FT0 oil", "Flow over weir", "Total liquid level change rate"* **Child of nodes**: *"FT1 given level LT1.CV opening"* **Type of node:** none **Input/output of the BBN:** Output	1: CV 0, 2: CV (0-20], 3: CV (20-40], 4: CV (40-60), 5: CV [60-80), 6: CV [80-100), 7: CV 100
"Water level change rate"	Rule 5 was used to create this node. It is used to model the water level change rate. The probability table is used to represent the certainty of water level change rate, given the states of the nodes *"Water inflow"* and *"Water outflow"*. For example, if the *"Water inflow"* node is in the state *"3: CV (20-40]"* and the *"Water outflow"* is in the state *"2: CV (0-20]"*, the state of the node *"Water level change rate"* will be either *"8: 0 < CV <= +20"* or *"9: +20 < CV <= +40"*. **Parent of nodes**: *"LT1 water level change rate"* **Child of nodes**: *"Water outflow", "Water inflow"* **Type of node:** none **Input/output of the BBN:** none	1: CV -100, 2: -100 < CV <= -80, 3: -80 < CV <= -60, 4: -60 < CV < -40, 5: -40 <= CV < -20, 6: -20 <= CV < 0, 7: CV 0, 8: 0 < CV <= +20, 9: +20 < CV <= +40, 10: +40 < CV < +60, 11: +60 <= CV < +80, 12: +80 <= CV < +100, 13: CV +100
"FT0 water –	Rule 6 was used to create this node. It is used to input the difference of the readings from flow	1: CV -100,

Node label	Description	Node states
FT1"	transmitter FT0 about the water inflow and the readings from flow transmitter FT1 about the water outflow. The probability table of this node is used to model the failure effects of flow transmitters FT0 and FT1 on the information about the water inflow and outflow difference. **Parent of nodes**: none **Child of nodes**: *"Water inflow", "Water outflow", "FT0 failure", "FT1 failure"* **Type of node:** Information node **Input/output of the BBN:** Output	2: -100 < CV <= -80, 3: -80 < CV <= -60, 4: -60 < CV < -40, 5: -40 <= CV < -20, 6: -20 <= CV < 0, 7: CV 0, 8: 0 < CV <= +20, 9: +20 < CV <= +40, 10: +40 < CV < +60, 11: +60 <= CV < +80, 12: +80 <= CV < +100, 13: CV +100
"LT1 water level change rate"	Rule 7 was used to create this node. It is used to input the readings from transmitter LT1 about the water level change rate. The probability table of this node is used to model the failure effects of transmitter LT1 on the information that it is showing about the water level change rate. **Parent of nodes**: none **Child of nodes**: *"Water level change rate", "LT1 failure"* **Type of node:** Information node **Input/output of the BBN:** Output	1: CV -100, 2: -100 < CV <= -80, 3: -80 < CV <= -60, 4: -60 < CV < -40, 5: -40 <= CV < -20, 6: -20 <= CV < 0, 7: CV 0, 8: 0 < CV <= +20, 9: +20 < CV <= +40, 10: +40 < CV < +60, 11: +60 <= CV < +80, 12: +80 <= CV < +100, 13: CV +100
"FT0 water – FT1 + FT0 oil"	This node was created as explained in the Step 1 of this section. It is used to input the difference of the readings from flow transmitter FT0 about the water inflow and the readings from flow transmitters FT1 about the water outflow and FT0 about the oil outflow. The probability table of this node is used to model the failure effects of flow transmitters FT0 and FT1 on the information about the liquid difference. **Parent of nodes**: none **Child of nodes**: *"Water inflow", "Water outflow", "Oil inflow", "FT0 failure", "FT1 failure"* **Type of node:** Information node **Input/output of the BBN:** Output	1: CV -100, 2: -100 < CV <= -80, 3: -80 < CV <= -60, 4: -60 < CV < -40, 5: -40 <= CV < -20, 6: -20 <= CV < 0, 7: CV 0, 8: 0 < CV <= +20, 9: +20 < CV <= +40,

Node label	Description	Node states
		10: +40 < CV < +60, 11: +60 <= CV < +80, 12: +80 <= CV < +100, 13: CV +100, 14: CV > +100
"Total liquid level"	This node was created as explained in the Step 2 of this section. It is used to model the total liquid level in the separation section. **Parent of nodes**: *"Flow over weir", "Total liquid level change rate", "LT1 total liquid level"* **Child of nodes**: none **Type of node:** none **Input/output of the BBN:** none	1: Equal with weir, 2: Below, + 0 < CV <= 20 3: Below, + 20 < CV <= 40, 4: Below, + 40 < CV < 60, 5: Below, + 60 <= CV < 80, 6: Below, + 80 <= CV < 100, 7: Below, + CV 100, 8: Below, > + CV 100
"LT1 total liquid level"	This node was created as explained in the Step 3 of this section. It is used to input the readings from transmitter LT1 about the total liquid level in the separation section. The probability table of this node is used to model the failure effects of transmitter LT1 on the information that it is showing about the total liquid level. **Parent of nodes**: none **Child of nodes**: *"Total liquid level", "LT1 failure"* **Type of node:** Information node **Input/output of the BBN:** Input	1: Equal with weir, 2: Below, + 0 < CV <= 20 3: Below, + 20 < CV <= 40, 4: Below, + 40 < CV < 60, 5: Below, + 60 <= CV < 80, 6: Below, + 80 <= CV < 100, 7: Below, + CV 100, 8: Below, > + CV 100
"Flow over weir"	This node was created as explained in the Step 4 of this section. It is used to model the flow over weir of the liquid from the separation section. The flow over weir depends on the total liquid level in the separation section and the incoming and outflowing liquid difference. **Parent of nodes**: none **Child of nodes**: *"Water inflow", "Water outflow", "Oil inflow", "Total liquid level"* **Type of node:** none **Input/output of the BBN:** Output	1: CV 0, 2: CV (0-20], 3: CV (20-40], 4: CV (40-60), 5: CV [60-80), 6: CV [80-100), 7: CV 100, 8: CV > 100
"Total liquid level change rate"	This node was created as explained in the Step 5 of this section. It is used to model the change rate of total liquid level. The probability table is used to represent the certainty of total liquid level change rate, given the states of the nodes *"Total liquid level"*, *"Water inflow", "Water outflow"* and *"Oil inflow"* **Parent of nodes**: *"LT1 total liquid level change rate"* **Child of nodes**: *"Water inflow", "Water outflow", "Oil inflow", "Total liquid level"*	1: CV -100, 2: -100 < CV <= -80, 3: -80 < CV <= -60, 4: -60 < CV < -40, 5: -40 <= CV < -20,

Node label	Description	Node states
	Type of node: none **Input/output of the BBN:** Output	6: -20 <= CV < 0, 7: CV 0, 8: 0 < CV <= +20, 9: +20 < CV <= +40, 10: +40 < CV < +60, 11: +60 <= CV < +80, 12: +80 <= CV < +100, 13: CV +100, 14: CV > +100
"LT1 total liquid level change rate"	This node was created as explained in the Step 6 of this section. It is used to input the readings from transmitter LT1 about the total liquid level change rate. The probability table of this node is used to model the failure effects of transmitter LT1 on the information that it is showing about the total liquid level change rate. **Parent of nodes**: none **Child of nodes**: *"Total liquid level change rate", "LT1 failure"* **Type of node:** Information node **Input/output of the BBN:** Output	1: CV -100, 2: -100 < CV <= -80, 3: -80 < CV <= -60, 4: -60 < CV < -40, 5: -40 <= CV < -20, 6: -20 <= CV < 0, 7: CV 0, 8: 0 < CV <= +20, 9: +20 < CV <= +40, 10: +40 < CV < +60, 11: +60 <= CV < +80, 12: +80 <= CV < +100, 13: CV +100, 14: CV > +100

Table 10.3. Summary of the BBN *"Separated oil section"*

Node name	Description	Node states
"Oil inflow"	Rule 1 was used to create this node. It represents the flow of oil from the separation section. **Parent of nodes**: *"Oil level change rate"* **Child of nodes**: none **Type of node:** none **Input/output of the BBN:** Input	1: CV 0, 2: CV (0-20], 3: CV (20-40], 4: CV (40-60), 5: CV [60-80), 6: CV [80-100),

Node name	Description	Node states
		7: CV 100, 8: CV > 100
"FT2 given level LT2"	Rule 4 was used to create this node. This is an instance node to model the oil level control loop. **Inputs of node**: *"LT info", "LT failure", "LC failure", "CV failure", "FT failure"* **Outputs of node**: *"FT info", "CV opening"* **Type of node:** Instance node **Input/output of the BBN:** none	-
"LT2 info"	Rule 4.a was used to create this node. It represents the oil level information transformed into a PI controller output command. **Parent of nodes**: *"FT2 given level LT2.LT info"* **Child of nodes**: none **Type of node:** Information node **Input/output of the BBN:** Input	1: CV 0, 2: CV (0-20], 3: CV (20-40], 4: CV (40-60), 5: CV [60-80), 6: CV [80-100), 7: CV 100
"LT2 failure"	Rule 4.b was used to create this node. It represents the condition of the transmitter LT2. The first state of the node represents the non-failed condition (LT W), while the second state represents the failure mode – failed stuck (LT FS). This node is linked to any nodes, which represent the information shown by the LT2 transmitter, so that the failure effects of specific failure modes could be modelled. **Parent of nodes**: *"FT2 given level LT2.LT failure", "LT2 oil level change rate"* **Child of nodes**: none **Type of node:** Condition node **Input/output of the BBN:** Input	1: LT W, 2: LT FS,
"LC2 failure"	Rule 4.c was used to create this node. It represents the condition of the controller LC2. First state of the node represents the non-failed condition (LC W), while the remaining states represent two failure modes – failed low (LC FL) and failed high (LC FH). This node is linked to any nodes, which represent the command that the controller LC2 sends to the valve CV2, so that the failure effects of specific failure modes could be modelled. **Parent of nodes**: *"FT2 given level LT2.LC failure"* **Child of nodes**: none **Type of node:** Condition node **Input/output of the BBN:** Input	1: LC W, 2: LC FL, 3: LC FH
"CV2 failure"	Rule 4.d was used to create this node. It represents the condition of the valve CV2. First state of the node represents the non-failed condition (CV W), while the remaining states represent two failure modes – failed closed (CV FC) and failed opened (CV FO). This node is linked to any nodes, which represent	1: CV W, 2: CV FC, 3: CV FO

Node name	Description	Node states
	the opening of valve CV2, so that the failure effects of specific failure modes could be modelled. **Parent of nodes**: *"FT2 given level LT2.CV failure"* **Child of nodes**: none **Type of node:** Condition node **Input/output of the BBN:** Input	
"FT2 failure"	Rule 4.e was used to create this node. It represents the condition of the flow transmitter FT2. First state of the node represents the non-failed condition (FT W), while the second state represents the failure mode – failed stuck (FT FS). This node is linked to any nodes, which represent the information shown by the flow transmitter FT2, so that the failure effects of specific failure modes could be modelled. **Parent of nodes**: *"FT2 given level LT2.FT failure"* **Child of nodes**: none **Type of node:** Condition node **Input/output of the BBN:** Input	1: FT W, 2: FT FS
"FT2 info"	Rule 4.f was used to create this node. It is used to input the readings from the flow transmitter FT2 about the oil outflow from the separator. **Parent of nodes**: none **Child of nodes**: *"FT2 given level LT2.FT info"* **Type of node:** Information node **Input/output of the BBN:** Output	1: CV 0, 2: CV (0-20], 3: CV (20-40], 4: CV (40-60), 5: CV [60-80), 6: CV [80-100), 7: CV 100
"Oil outflow"	Rule 4.g was used to create this node. Node is used to obtain the information from the output node *"CV opening"* of the instance node *"FT2 given level LT2"*. It is further used to model the oil level change rate in the separated oil section. **Parent of nodes**: *"Oil level change rate"* **Child of nodes**: *"FT2 given level LT2.CV opening"* **Type of node:** none **Input/output of the BBN:** none	1: CV 0, 2: CV (0-20], 3: CV (20-40], 4: CV (40-60), 5: CV [60-80), 6: CV [80-100), 7: CV 100
"Oil level change rate"	Rule 5 was used to create this node. It is used to model the oil level change rate. The conditional probability table is used to represent the certainty of oil level change rate, given the states of the nodes *"Oil inflow"* and *"Oil outflow"*. For example, if the *"Oil inflow"* node is in the state *"3: CV (20-40]"* and the *"Oil outflow"* is in the state *"2: CV (0-20]"*, the state of the node *"Oil level change rate"* will be either *"8: 0 < CV <= +20"* or *"9: +20 < CV <= +40"*. **Parent of nodes**: *"LT2 oil level change rate"*	1: CV -100, 2: -100 < CV <= -80, 3: -80 < CV <= -60, 4: -60 < CV < -40, 5: -40 <= CV < -20, 6: -20 <= CV < 0, 7: CV 0,

Node name	Description	Node states
	Child of nodes: *"Oil inflow", "Oil outflow"* **Type of node:** none **Input/output of the BBN:** Output	8: 0 < CV <= +20, 9: +20 < CV <= +40, 10: +40 < CV < +60, 11: +60 <= CV < +80, 12: +80 <= CV < +100, 13: CV +100
"LT2 oil level change rate"	Rule 7 was used to create this node. It is used to input the readings from level transmitter LT2 about the oil level change rate. The conditional probability table of this node is used to model the failure effects of level transmitter LT2 on the information that it is showing about the oil level change rate. **Parent of nodes**: Output **Child of nodes**: *"Oil level change rate", "LT2 failure"* **Type of node:** Information node **Input/output of the BBN:** none	1: CV -100, 2: -100 < CV <= -80, 3: -80 < CV <= -60, 4: -60 < CV < -40, 5: -40 <= CV < -20, 6: -20 <= CV < 0, 7: CV 0, 8: 0 < CV <= +20, 9: +20 < CV <= +40, 10: +40 < CV < +60, 11: +60 <= CV < +80, 12: +80 <= CV < +100, 13: CV +100

Table 10.4. Summary of the BBN *"Gas section"*

Node name	Description	Node states
"Gas inflow"	Rule 1 was used to create this node. It represents the inflowing gas into the separator. If certain knowledge is available about the inflowing gas, this can be represented by increasing the prior probabilities of specific states of the node. It is assumed that different inflows have equal probabilities, thus each state of the node has the same prior probability. **Parent of nodes**: *"FT0 gas info"*, *"Gas level change rate"* **Child of nodes**: none **Type of node:** none **Input/output of the BBN:** none	1: CV 0, 2: CV (0-20], 3: CV (20-40], 4: CV (40-60), 5: CV [60-80), 6: CV [80-100), 7: CV 100
"FT0 failure"	Rule 2 was used to create this node. It represents the condition of flow transmitter FT0. First state of the node represents the non-failed condition (FT W), while the second state represents the failure mode –	1: FT W, 2: FT FS

Node name	Description	Node states
	failed stuck (FT FS). This node is linked to any nodes, which represent the information shown by the flow transmitter FT0, so that the failure effects of specific failure modes could be modelled. **Parent of nodes**: *"FT0 gas inflow info", "FT0 gas – FT3"* **Child of nodes**: none **Type of node:** Condition node **Input/output of the BBN:** Input	
"FT0 gas inflow info"	Rule 3 was used to create this node. It is used to input the readings from the flow transmitter FT0 about the gas flow into the separator. The probability table of this node is used to model the failure effects of flow transmitter FT0 on the information that it is showing about the gas flow. **Parent of nodes**: none **Child of nodes**: *"FT0 failure", "Gas inflow"* **Type of node:** Information node **Input/output of the BBN:** Output	1: CV 0, 2: CV (0-20], 3: CV (20-40], 4: CV (40-60), 5: CV [60-80), 6: CV [80-100), 7: CV 100
"FT3 given level LT3"	Rule 4 was used to create this node. This is an instance node to model the pressure level control loop. **Inputs of node**: *"LT info", "LT failure", "LC failure", "CV failure", "FT failure"* **Outputs of node**: *"FT info", "CV opening"* **Type of node:** Instance node **Input/output of the BBN:** none	-
"LT3 info"	Rule 4.a was used to create this node. It represents the pressure level information transformed into a PI controller output command. **Parent of nodes**: *"FT3 given level LT3.LT info"* **Child of nodes**: none **Type of node:** Information node **Input/output of the BBN:** Input	1: CV 0, 2: CV (0-20], 3: CV (20-40], 4: CV (40-60), 5: CV [60-80), 6: CV [80-100), 7: CV 100
"LT3 failure"	Rule 4.b was used to create this node. It represents the condition of the transmitter LT3. The first state of the node represents the non-failed condition (LT W), while the second state represents the failure mode – failed stuck (LT FS). This node is linked to any nodes, which represent the information shown by the LT3 transmitter, so that the failure effects of specific failure modes could be modelled. **Parent of nodes**: *"FT3 given level LT3.LT failure", "LT3 pressure level change rate"* **Child of nodes**: none **Type of node:** Condition node **Input/output of the BBN:** Input	1: LT W, 2: LT FS,
"LC3 failure"	Rule 4.c was used to create this node. It represents the condition of the controller LC3. First state of the	1: LC W,

Node name	Description	Node states
	node represents the non-failed condition (LC W), while the remaining states represent two failure modes – failed low (LC FL) and failed high (LC FH). This node is linked to any nodes, which represent the command that the controller LC3 sends to the valve CV3, so that the failure effects of specific failure modes could be modelled. **Parent of nodes**: *"FT3 given level LT3.LC failure"* **Child of nodes**: none **Type of node:** Condition node **Input/output of the BBN:** Input	2: LC FL, 3: LC FH
"CV3 failure"	Rule 4.d was used to create this node. It represents the condition of the valve CV3. First state of the node represents the non-failed condition (CV W), while the remaining states represent two failure modes – failed closed (CV FC) and failed opened (CV FO). This node is linked to any nodes, which represent the opening of valve CV3, so that the failure effects of specific failure modes could be modelled. **Parent of nodes**: *"FT3 given level LT3.CV failure"* **Child of nodes**: none **Type of node:** Condition node **Input/output of the BBN:** Input	1: CV W, 2: CV FC, 3: CV FO
"FT3 failure"	Rule 4.e was used to create this node. It represents the condition of the flow transmitter FT3. First state of the node represents the non-failed condition (FT W), while the second state represents the failure mode – failed stuck (FT FS). This node is linked to any nodes, which represent the information shown by the flow transmitter FT3, so that the failure effects of specific failure modes could be modelled. **Parent of nodes**: *"FT3 given level LT3.FT failure", "FT0 gas – FT3"* **Child of nodes**: none **Type of node:** Condition node **Input/output of the BBN:** Input	1: FT W, 2: FT FS
"FT3 info"	Rule 4.f was used to create this node. It is used to input the readings from the flow transmitter FT3 about the gas outflow from the separator. **Parent of nodes**: none **Child of nodes**: *"FT3 given level LT3.FT info"* **Type of node:** Information node **Input/output of the BBN:** Output	1: CV 0, 2: CV (0-20], 3: CV (20-40], 4: CV (40-60), 5: CV [60-80), 6: CV [80-100), 7: CV 100
"Gas outflow"	Rule 4.g was used to create this node. It is used to obtain the information from the output node *"CV opening"* of the instance node *"FT3 given level LT3"*. It is further used to model the gas level change rate in the gas section.	1: CV 0, 2: CV (0-20], 3: CV (20-40],

Node name	Description	Node states
	Parent of nodes: *"Gas level change rate"* **Child of nodes**: *"FT3 given level LT3.CV opening"* **Type of node:** none **Input/output of the BBN:** none	4: CV (40-60), 5: CV [60-80), 6: CV [80-100), 7: CV 100
"Gas level change rate"	Rule 5 was used to create this node. It is used to model the gas level change rate. The conditional probability table is used to represent the certainty of gas level change rate, given the states of the nodes *"Incoming gas"* and *"Gas outflow"*. For example, if the *"Incoming gas"* node is in the state *"3: CV (20-40]"* and the *"Gas outflow"* is in the state *"2: CV (0-20]"*, the state of the node *"Gas level change rate"* will be either *"8: 0 < CV <= +20"* or *"9: +20 < CV <= +40"*. **Parent of nodes**: *"Pressure level change rate", "FT0 gas – FT3"* **Child of nodes**: *"Gas outflow", "Incoming gas"* **Type of node:** none **Input/output of the BBN:** none	1: CV -100, 2: -100 < CV <= -80, 3: -80 < CV <= -60, 4: -60 < CV < -40, 5: -40 <= CV < -20, 6: -20 <= CV < 0, 7: CV 0, 8: 0 < CV <= +20, 9: +20 < CV <= +40, 10: +40 < CV < +60, 11: +60 <= CV < +80, 12: +80 <= CV < +100, 13: CV +100
"FT0 gas – FT3"	Rule 6 was used to create this node. It is used to input the difference of the readings from flow transmitter FT0 about the gas inflow and the readings from flow transmitter FT3 about the gas outflow. The conditional probability table of this node is used to model the failure effects of flow transmitters FT0 and FT3 on the information about the gas inflow and outflow difference. **Parent of nodes**: none **Child of nodes**: *"Gas inflow", "Gas outflow", "FT0 failure", "FT3 failure"* **Type of node:** Information node **Input/output of the BBN:** Output	1: CV -100, 2: -100 < CV <= -80, 3: -80 < CV <= -60, 4: -60 < CV < -40, 5: -40 <= CV < -20, 6: -20 <= CV < 0, 7: CV 0, 8: 0 < CV <= +20, 9: +20 < CV <= +40, 10: +40 < CV < +60, 11: +60 <= CV < +80, 12: +80 <= CV < +100, 13: CV +100
"Change of volume for gas"	This node was created as explained in the Step 1 of this section. **Parent of nodes**: *"Pressure level change rate"* **Child of nodes**: none **Type of node:** none	1: CV -100, 2: -100 < CV <= -80, 3: -80 < CV <= -60, 4: -60 < CV < -40,

Node name	Description	Node states
	Input/output of the BBN: Input	5: -40 <= CV < -20, 6: -20 <= CV < 0, 7: CV 0, 8: 0 < CV <= +20, 9: +20 < CV <= +40, 10: +40 < CV < +60, 11: +60 <= CV < +80, 12: +80 <= CV < +100, 13: CV +100
"Pressure level change rate"	This node was created as explained in the Step 2 of this section. **Parent of nodes**: *"LT3 pressure level change rate"* **Child of nodes**: *"Change of volume for gas", "Gas level change rate"* **Type of node:** none **Input/output of the BBN:** none	1: CV < -100 2: CV -100 3: -100 < CV <= -80 4: -80 < CV <= -60 5: -60 < CV < -40 6: -40 <= CV < -20 7: -20 <= CV < 0 8: CV 0 9: 0 < CV <= +20 10: +20 < CV <= +40 11: +40 < CV < +60 12: +60 <= CV < +80 13: +80 <= CV < +100 14: CV +100 15: CV > +100
"LT3 pressure level change rate"	This node was created as explained in the Step 3 of this section. **Parent of nodes**: none **Child of nodes**: *"Pressure level change rate", "LT3 failure"* **Type of node:** Information node **Input/output of the BBN:** Output	1: CV < -100 2: CV -100 3: -100 < CV <= -80 4: -80 < CV <= -60 5: -60 < CV < -40 6: -40 <= CV < -20 7: -20 <= CV < 0 8: CV 0 9: 0 < CV <= +20 10: +20 < CV <= +40

Node name	Description	Node states
		11: +40 < CV < +60 12: +60 <= CV < +80 13: +80 <= CV < +100 14: CV +100 15: CV > +100

Table 10.5. Summary of the BBN *"Single time slice"*

Node name	Description	Node states
"Separation section"	This is an instance node of BBN "Separation section". **Inputs of node**: *"LT1 info", "FT0 failure", "LT1 failure", "LC1 failure", "CV1 failure", "FT1 failure"* **Outputs of node**: *"FT0 water inflow info", "FT0 oil inflow info", "FT1 info", "FT0 water – FT1", "LT1 water level change rate", "FT0 water – FT1 + FT0 oil", "LT1 total liquid level", "LT1 total liquid level", "LT1 total liquid level change rate", "Total liquid level change rate", "Flow over weir", "Water inflow", "Water outflow", "Oil inflow"* **Type of node:** Instance node **Input/output of the BBN:** none	-
"Separated oil section"	This is an instance node of BBN "*Separated oil section*". **Inputs of node**: *"LT2 info", "LT2 failure", "LC2 failure", "CV2 failure", "FT2 failure", "Oil inflow"* **Outputs of node**: *"FT2 info", "LT2 oil level change rate", "Oil level change rate", "Oil outflow"* **Type of node:** Instance node **Input/output of the BBN:** none	-
"Gas section"	This is an instance node of BBN "*Gas section*". **Inputs of node**: *"Change of volume for gas"*, *"LT3 info", "LT3 failure", "LC3 failure", "CV3 failure", "FT3 failure", "FT0 failure"* **Outputs of node**: *"FT0 gas inflow info", "FT3 info", "FT0 gas – FT3", "LT3 pressure level change rate"* **Type of node:** Instance node **Input/output of the BBN:** none	-
"Flow over weir"	This node was created as explained in the Step 1 of this section. **Parent of nodes**: *"Separated oil section.Oil inflow"* **Child of nodes**: *"Separation section. Flow over weir"* **Type of node:** none	

Node name	Description	Node states
	Input/output of the BBN: none	
"Total liquid level change rate single"	This node was created as explained in the Step 2 of this section. **Parent of nodes**: *"LT1 total liquid level change + LT2 level change"* **Child of nodes**: *"Separation section.Total liquid level change rate"* **Type of node:** none **Input/output of the BBN:** none	
"Oil level change rate single"	This node was created as explained in the Step 3 of this section. **Parent of nodes**: *"Change of volume for gas"* **Child of nodes**: *"Separated oil section.Oil level change rate"* **Type of node:** none **Input/output of the BBN:** none	
"Change of volume for gas from FT"	This node was created as explained in the Step 5 of this section. **Parent of nodes**: *"Gas section.Change of volume for gas"* **Child of nodes**: *"Water inflow single", "Water outflow single", "Oil inflow single", "Oil outflow single"* **Type of node:** none **Input/output of the BBN:** none	1: CV -100 2: -100 < CV <= -80 3: -80 < CV <= -60 4: -60 < CV < -40 5: -40 <= CV < -20 6: -20 <= CV < 0 7: CV 0 8: 0 < CV <= +20 9: +20 < CV <= +40 10: +40 < CV < +60 11: +60 <= CV < +80 12: +80 <= CV < +100 13: CV +100
"FT0 water – FT1 + FT0 oil – FT2"	This node was created as explained in the Step 6 of this section. **Parent of nodes**: none **Child of nodes**: *"Water inflow single", "Water outflow single", "Oil inflow single", "Oil outflow single", "FT0 failure", "FT1 failure", "FT2 failure"* **Type of node:** Information node **Input/output of the BBN:** Output	
"LT1 total liquid level change + LT2 level change"	This node was created as explained in the Step 7 of this section. **Parent of nodes**: none **Child of nodes**: *"Total liquid level change rate single", "Oil level change rate single", "LT1 failure", "LT2 failure"*	

Node name	Description	Node states
	Type of node: Information node **Input/output of the BBN:** Output	
"FT0 varied"	This node was created as explained in the Step 8 of this section. **Parent of nodes**: none **Child of nodes**: *"FT0 failure"* **Type of node:** Information node **Input/output of the BBN:** Output	1: FT W, 2: FT FS
"FT1 varied"	This node was created as explained in the Step 8 of this section. **Parent of nodes**: none **Child of nodes**: *"FT1 failure"* **Type of node:** Information node **Input/output of the BBN:** Output	1: false, 2: true
"FT2 varied"	This node was created as explained in the Step 8 of this section. **Parent of nodes**: none **Child of nodes**: *"FT2 failure"* **Type of node:** Information node **Input/output of the BBN:** Output	1: false, 2: true
"FT3 varied"	This node was created as explained in the step Step 8 of this section. **Parent of nodes**: none **Child of nodes**: *"FT3 failure"* **Type of node:** Information node **Input/output of the BBN:** Output	1: false, 2: true

11 Appendix F

Table 11.1. Summary of undetected failures, when LT2 failed stuck failure was simulated together with another component failing, given oscillating inflows

Fault 1	LT2 FS											
Fault 2	FT0 FS	LT1 FS	CV1 FC	CV1 FO	FT1 FS	CV2 FC	CV2 FO	FT2 FS	LT3 FS	CV3 FC	CV3 FO	FT3 FS
Times **Fault 1** not detected	0	0	0	1	0	0	0	0	0	0	0	0
Times **Fault 2** not detected	0	7	0	0	1	18	19	37	7	6	4	10
Times **Fault 1** and **Fault 2** not detected	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.2. Summary of undetected failures, when LT3 failed stuck failure was simulated together with another component failing, given oscillating inflows

Fault 1	LT3 FS											
Fault 2	FT0 FS	LT1 FS	CV1 FC	CV1 FO	FT1 FS	LT2 FS	CV2 FC	CV2 FO	FT2 FS	CV3 FC	CV3 FO	FT3 FS
Times **Fault 1** not detected	0	0	0	0	0	0	1	0	0	0	0	0
Times **Fault 2** not detected	0	0	0	0	1	0	0	0	0	20	24	44
Times **Fault 1** and **Fault 2** not detected	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.3. Faults detected with BBN, when two faults have been inserted in the simulation model, when oscillating inflows were modelled

Fault 1	Fault 2	Faults detected																	
		FT0 FS	LT1 FS	LC1 FH	LC1 FL	CV1 FC	CV1 FO	FT1 FS	LT2 FS	LC2 FH	LC2 FL	CV2 FC	CV2 FO	FT2 FS	LT3 FS	LC3 FH	LC3 FL	CV3 FC	CV3 FO	FT3 FS
LT2 FS	FT0 FS	**67.38**	0.84						**83.03**								1.29	1.29		1.29
	LT1 FS		**68.71**						**84.68**		0.28	0.28		0.33		1.99	3.43	3.43	1.82	5.31
	CV1 FC		0.11		24.59	**86.78**			**86.49**		1.09	1.09		1.30		0.33	3.80	3.47	0.33	3.85
	CV1 FO		0.17	22.66			**86.72**		**82.16**							4.81	3.69	1.40	4.81	5.93
	FT1 FS		0.28	1.12	1.06	0.95	1.17	**72.07**	**84.50**							0.39	2.96	2.57	0.39	2.96
	CV2 FC		0.31						**88.92**		23.10	**71.36**				2.20	6.20	4.00	2.20	5.07
	CV2 FO		0.11						**84.89**	16.79			**68.94**	0.06						
	FT2 FS		0.11						**85.64**	0.67			0.67	**30.92**			3.35	3.35		3.69
	LT3 FS		0.17						**84.35**						**71.92**					
	CV3 FC		0.18						**81.62**								17.60	**79.89**		
	CV3 FO		0.17						**84.45**							21.94	1.45	2.13	**78.37**	2.46
	FT3 FS		0.17						**84.50**	2.96	1.45	1.68	2.74	4.47	0.11	2.74	5.76	6.32	2.85	**50.32**
LT3 FS	FT0 FS	**72.71**	0.69												**89.66**					
	LT1 FS		**79.72**												**87.53**					
	CV1 FC				25.60	**87.39**									**82.92**					
	CV1 FO			30.93			**90.90**								**86.06**					
	FT1 FS			1.20	1.17	1.17	1.20	**74.12**							**92.66**					
	LT2 FS		0.16						**84.79**						**85.98**					
	CV2 FC										23.46	**95.14**			**78.47**					
	CV2 FO									18.68			**93.62**		**75.26**					
	FT2 FS									0.22	0.33	0.33	0.22	**96.84**	**92.66**					
	CV3 FC														**93.63**		29.22	**57.52**		1.71
	CV3 FO														**93.10**	25.65			**48.77**	0.11
	FT3 FS														**92.91**					**8.16**

Table 11.4. Summary of undetected failures, when LT2 failed stuck failure was simulated together with another component failing, given decreasing inflows

Fault 1	LT2 FS											
Fault 2	FT0 FS	LT1 FS	CV1 FC	CV1 FO	FT1 FS	CV2 FC	CV2 FO	FT2 FS	LT3 FS	CV3 FC	CV3 FO	FT3 FS
Times **Fault 1** not detected	0	2	0	4	0	0	0	0	0	0	0	0
Times **Fault 2** not detected	23	20	0	0	0	12	13	32	7	11	9	17
Times **Fault 1** and **Fault 2** not detected	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.5. Summary of undetected failures, when LT3 failed stuck failure was simulated together with another component failing, given decreasing inflows

Fault 1	LT3 FS											
Fault 2	FT0 FS	LT1 FS	CV1 FC	CV1 FO	FT1 FS	LT2 FS	CV2 FC	CV2 FO	FT2 FS	CV3 FC	CV3 FO	FT3 FS
Times **Fault 1** not detected	0	1	2	3	1	2	3	5	1	2	0	1
Times **Fault 2** not detected	4	4	0	0	0	0	0	0	0	17	19	36
Times **Fault 1** and **Fault 2** not detected	0	1	0	0	0	0	0	0	0	0	0	0

Table 11.6. Faults detected with BBN, when two faults have been inserted in the simulation model, when decreasing inflows were modelled

Fault 1	**Fault 2**	**Faults detected**																	
		FT0 FS	LT1 FS	LC1 FH	LC1 FL	CV1 FC	CV1 FO	FT1 FS	LT2 FS	LC2 FH	LC2 FL	CV2 FC	CV2 FO	FT2 FS	LT3 FS	LC3 FH	LC3 FL	CV3 FC	CV3 FO	FT3 FS
LT2 FS	FT0 FS	**28.59**	2.07						**76.85**						0.52					
	LT1 FS		**61.17**						**75.03**								0.15	0.15		0.15
	CV1 FC		0.32		30.53	**100.00**			**78.87**						0.05	0.05	2.00	1.95	0.05	2.00
	CV1 FO		0.33	26.57			**99.79**	0.06	**70.14**							1.22	0.33	0.33	0.89	1.28
	FT1 FS		1.19	0.16			0.16	**82.19**	**78.31**						0.72	0.10	0.05	0.05	0.05	0.26
	CV2 FC		1.92		0.19	0.19		0.19	**84.85**		26.15	**71.58**		0.58		0.38	1.01	0.72	0.53	0.91
	CV2 FO		0.90						**77.30**	18.48			**72.09**	0.23	0.62	0.45	0.23	0.23	0.23	1.07
	FT2 FS		0.88						**80.03**		0.16	0.16		**29.74**	0.72	0.26	0.05	0.21	0.21	0.26
	LT3 FS		1.19						**78.31**						**65.85**					
	CV3 FC		1.27						**73.89**						0.22	0.06	22.17	**83.26**	0.06	0.17
	CV3 FO		1.23						**76.60**						0.32	22.38			**75.40**	
	FT3 FS		1.19						**78.31**		1.60	1.60		1.86	0.72	3.73	0.83	1.04	3.16	**61.93**
LT3 FS	FT0 FS	**59.47**	2.30												**90.29**					
	LT1 FS		**64.01**						0.30						**89.42**					
	CV1 FC				40.67	**100.00**			0.13						**86.18**					
	CV1 FO			34.78			**100.00**								**82.08**					
	FT1 FS							**92.33**							**90.16**					
	LT2 FS		1.12						**78.39**						**80.91**					
	CV2 FC										26.82	**98.19**		0.39	**77.34**					
	CV2 FO		0.12		0.12	0.12		0.18		20.53			**100.00**	0.18	**71.34**					
	FT2 FS									0.07	0.18	0.26	0.04	**96.55**	**90.16**					
	CV3 FC														**92.23**		34.58	**67.85**		0.59
	CV3 FO														**92.79**	29.02			**55.19**	0.22
	FT3 FS														**90.80**	1.20			1.20	**22.24**

Table 11.7. Summary of undetected failures, when LT2 failed stuck failure was simulated together with another component failing, given increasing inflows

Fault 1	LT2 FS											
Fault 2	FT0 FS	LT1 FS	CV1 FC	CV1 FO	FT1 FS	CV2 FC	CV2 FO	FT2 FS	LT3 FS	CV3 FC	CV3 FO	FT3 FS
Times **Fault 1** not detected	0	2	0	3	0	0	1	0	0	0	0	0
Times **Fault 2** not detected	18	4	0	0	0	12	13	30	6	7	7	13
Times **Fault 1** and **Fault 2** not detected	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.8. Summary of undetected failures, when LT3 failed stuck failure was simulated together with another component failing, given increasing inflows

Fault 1	LT3 FS											
Fault 2	FT0 FS	LT1 FS	CV1 FC	CV1 FO	FT1 FS	LT2 FS	CV2 FC	CV2 FO	FT2 FS	CV3 FC	CV3 FO	FT3 FS
Times **Fault 1** not detected	1	1	0	4	1	1	1	3	1	0	1	1
Times **Fault 2** not detected	4	6	0	0	0	0	0	0	0	19	18	38
Times **Fault 1** and **Fault 2** not detected	1	1	0	0	0	0	0	0	0	0	1	1

Table 11.9. Faults detected with BBN, when two faults have been inserted in the simulation model, when increasing inflows were modelled

Fault 1	Fault 2	Faults detected																	
		FT0 FS	LT1 FS	LC1 FH	LC1 FL	CV1 FC	CV1 FO	FT1 FS	LT2 FS	LC2 FH	LC2 FL	CV2 FC	CV2 FO	FT2 FS	LT3 FS	LC3 FH	LC3 FL	CV3 FC	CV3 FO	FT3 FS
LT2 FS	FT0 FS	**21.77**	0.61						**76.75**						0.61					
	LT1 FS		**68.46**						**71.37**								0.64	0.64		0.64
	CV1 FC				28.65	**100.00**			**74.33**						0.05	0.27	1.35	1.19	0.27	1.41
	CV1 FO			29.02			**99.82**	0.05	**69.49**						0.38	0.86	0.21	0.38	0.86	0.86
	FT1 FS		0.31		0.05	0.05		**91.91**	**79.54**											
	CV2 FC		0.05						**81.89**		23.85	**72.20**		0.61	0.10					
	CV2 FO								**77.91**	21.46			**74.39**	0.32		0.54	0.27	0.27	0.27	1.35
	FT2 FS								**80.56**		0.05	0.05		**28.93**						
	LT3 FS								**79.53**						**69.90**					
	CV3 FC								**74.88**								23.99	**89.28**		
	CV3 FO								**78.62**							27.15			**87.39**	
	FT3 FS								**79.92**	1.38	0.66	0.41	1.17	1.43		1.38	1.48	1.38	1.63	**72.78**
LT3 FS	FT0 FS	**53.21**							3.44						**85.34**					
	LT1 FS		**68.04**						1.68						**86.60**					
	CV1 FC				36.11	**100.00**			0.10						**83.48**					
	CV1 FO			38.07			**99.88**	0.14							**80.25**					
	FT1 FS			0.04			0.04	**96.17**	4.15						**88.34**					
	LT2 FS								**79.30**						**79.95**					
	CV2 FC		0.06								23.56	**98.32**		0.46	**70.89**					
	CV2 FO									24.04			**99.76**		**70.16**					
	FT2 FS								2.91	0.15	0.34	0.19	0.22	**96.02**	**88.27**					
	CV3 FC								3.50						**89.66**		27.76	**56.46**		0.27
	CV3 FO								5.17						**90.92**	36.93			**70.22**	0.29
	FT3 FS								4.18						**89.14**					**22.99**

Table 11.10. Summary of undetected failures, when LT2 failed stuck failure was simulated together with another component failing, given random inflows

Fault 1	LT2 FS											
Fault 2	FT0 FS	LT1 FS	CV1 FC	CV1 FO	FT1 FS	CV2 FC	CV2 FO	FT2 FS	LT3 FS	CV3 FC	CV3 FO	FT3 FS
Times **Fault 1** not detected	0	0	0	2	0	0	0	0	0	0	0	0
Times **Fault 2** not detected	18	7	0	0	1	21	21	42	4	16	12	31
Times **Fault 1** and **Fault 2** not detected	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.11. Summary of undetected failures, when LT3 failed stuck failure was simulated together with another component failing, given random inflows

Fault 1	LT3 FS											
Fault 2	FT0 FS	LT1 FS	CV1 FC	CV1 FO	FT1 FS	LT2 FS	CV2 FC	CV2 FO	FT2 FS	CV3 FC	CV3 FO	FT3 FS
Times **Fault 1** not detected	0	0	0	0	0	0	0	0	0	0	0	0
Times **Fault 2** not detected	0	0	0	0	1	0	0	0	0	25	20	43
Times **Fault 1** and **Fault 2** not detected	0	0	0	0	0	0	0	0	0	0	0	0

Table 11.12. Faults detected with BBN, when two faults have been inserted in the simulation model, when random inflows were modelled

Fault 1	**Fault 2**	**Faults detected**																	
		FT0 FS	LT1 FS	LC1 FH	LC1 FL	CV1 FC	CV1 FO	FT1 FS	LT2 FS	LC2 FH	LC2 FL	CV2 FC	CV2 FO	FT2 FS	LT3 FS	LC3 FH	LC3 FL	CV3 FC	CV3 FO	FT3 FS
LT2 FS	FT0 FS	**44.08**							**86.12**											
	LT1 FS	2.51	**64.86**						**86.64**								0.48	0.48		0.48
	CV1 FC	1.56			33.27	**99.70**		0.05	**90.06**								0.61	0.61		0.61
	CV1 FO	2.63		24.43			**99.77**		**75.52**		0.11	0.11		0.11		0.57			0.57	0.57
	FT1 FS	2.21	0.11	0.11	1.33	1.33	0.11	**90.82**	**86.28**											
	CV2 FC	2.83							**90.82**		21.44	**62.16**		0.15						
	CV2 FO	0.40							**86.51**	16.19			**65.44**							
	FT2 FS	1.88							**87.38**					**14.90**						
	LT3 FS	0.66							**86.44**						**63.06**					
	CV3 FC	0.53							**83.65**								15.98	**71.20**		
	CV3 FO	1.95							**86.15**							18.64			**70.23**	
	FT3 FS	4.09							**85.80**	0.06			0.06	0.06	0.17	3.09	3.04	2.82	3.09	**33.27**
LT3 FS	FT0 FS	**68.49**													**94.84**					
	LT1 FS	1.64	**76.81**						0.62						**94.52**					
	CV1 FC	0.13			42.52	**99.80**									**93.68**					
	CV1 FO			33.68			**99.85**								**90.44**					
	FT1 FS	0.15		0.07	0.18	0.18	0.07	**96.80**							**94.77**					
	LT2 FS	0.73							**85.00**						**89.52**					
	CV2 FC										24.32	**96.62**		0.06	**86.39**					
	CV2 FO									18.70			**93.07**	0.06	**81.93**					
	FT2 FS	0.99								0.99	0.84	0.84	1.10	**94.52**	**94.65**					
	CV3 FC														**95.09**		24.08	**47.07**		0.04
	CV3 FO														**96.08**	32.22			**61.27**	0.14
	FT3 FS														**95.55**		1.91	1.91		**11.10**