Early Fault Detection in Industrial Plants
Is it possible to decrease unscheduled downtime?

D.W.B. Bender
Early Fault Detection in Industrial Plants

Is it possible to decrease unscheduled downtime?

MASTER OF SCIENCE THESIS

For the degree of Master of Science in Systems and Control at Delft University of Technology

D.W.B. Bender

27th June, 2016

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology
The work in this thesis was supported by Royal Dutch Shell. Its cooperation is hereby gratefully acknowledged.
Modern industrial plants contain enormous numbers of sensors which, in turn, generate enormous amounts of process and diagnostic variable measurements. All this generated data is stored in a Data Historian database and then left untouched. This report evaluates whether there is useful information amongst this unused data, and if so, how this information can best be used to increase the reaction time of plant operators. This is done by examining the application of regression methods to make early faults detection possible. The simulations are performed using historic process data from a crude distiller unit at the Shell Pernis Refinery. Datasets representing both normal and faulty operations are taken from two different subsystems of the crude distiller unit. The output datasets have irregular sampling times that are larger than the input variable datasets so this potential problem is solved by using a linear interpolation to estimate the missing values in the output datasets. The processes in the subsystems are modelled using finite impulse response (FIR) models. Five different regression methods are used to identify these models. This report concludes firstly that the ordinary least squares and ridge regression methods can be used to construct accurate out-of-sample prediction models of key process variables; and secondly, that this can be done without prior process knowledge or extensive process specific analysis.
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preface</td>
<td>vii</td>
</tr>
<tr>
<td>1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>2 Fundamentals</td>
<td>5</td>
</tr>
<tr>
<td>2-1 Process monitoring</td>
<td>5</td>
</tr>
<tr>
<td>2-1-1 Process monitoring definitions</td>
<td>6</td>
</tr>
<tr>
<td>2-1-2 Fault detection methods</td>
<td>7</td>
</tr>
<tr>
<td>2-1-3 Fault diagnosis</td>
<td>10</td>
</tr>
<tr>
<td>2-1-4 Early fault detection</td>
<td>10</td>
</tr>
<tr>
<td>3 Case study</td>
<td>13</td>
</tr>
<tr>
<td>3-1 Crude Distiller X1</td>
<td>13</td>
</tr>
<tr>
<td>3-2 Depentanizer</td>
<td>14</td>
</tr>
<tr>
<td>3-2-1 Quality variable QTXX1</td>
<td>15</td>
</tr>
<tr>
<td>3-2-2 Quality estimator QEXX1</td>
<td>17</td>
</tr>
<tr>
<td>3-2-3 Process variables TTX01, PTXX1, XCXX1, and FCXX3</td>
<td>22</td>
</tr>
<tr>
<td>4 Model analysis</td>
<td>23</td>
</tr>
<tr>
<td>4-1 Performance evaluation</td>
<td>24</td>
</tr>
<tr>
<td>4-1-1 Mean squared prediction error</td>
<td>25</td>
</tr>
<tr>
<td>4-2 Dynamic model</td>
<td>26</td>
</tr>
<tr>
<td>4-3 Variable types and terminology</td>
<td>28</td>
</tr>
<tr>
<td>4-4 Ordinary least squares</td>
<td>28</td>
</tr>
<tr>
<td>4-4-1 Concatenated variables</td>
<td>30</td>
</tr>
<tr>
<td>4-5 Dual and irregular sampling rate</td>
<td>30</td>
</tr>
<tr>
<td>4-6 Comparison identified method and quality estimator</td>
<td>32</td>
</tr>
</tbody>
</table>
5 Regression Methods
  5-1 Regularisation methods ........................................... 35
    5-1-1 Ridge regression ........................................... 35
    5-1-2 LASSO method .............................................. 36
  5-2 Methods using derived inputs .................................. 36
    5-2-1 Principal components regression ......................... 37
    5-2-2 Partial least squares regression ......................... 38
  5-3 Data pre-processing ............................................. 38
    5-3-1 Data normalisation ....................................... 38
    5-3-2 Removing erroneous measurement data .................. 39

6 Method Analysis
  6-1 Regularisation methods ......................................... 41
    6-1-1 Ridge regression ........................................... 41
    6-1-2 LASSO ...................................................... 42
  6-2 Methods using derived inputs ................................ 43
    6-2-1 Principal components regression ......................... 43
    6-2-2 Partial least squares regression ......................... 44

7 Results
  7-1 Normal operations dataset .................................... 46
    7-1-1 Concatenated variables .................................. 51
  7-2 Abnormal operations dataset ................................ 53
    7-2-1 Concatenated variables .................................. 58

8 Validation
  8-1 Deisopentanizer ................................................. 61
    8-1-1 Prediction results ....................................... 62
    8-1-2 Regression method analysis ............................... 65
    8-1-3 Concatenated variables .................................. 68

9 Concluding remarks .................................................. 71

A Case study
  A-1 Depentanizer ..................................................... 76
  A-2 Deisopentanizer .................................................. 77

B Model Analysis
  B-1 Results of OLS method with four input variables ........... 79
    B-1-1 Interpolated output results .............................. 79
    B-1-2 Measured output results .................................. 81
    B-1-3 Comparison ................................................. 82
  B-2 Estimated parameters of OLS with four input variables ... 83
    B-2-1 Rolling window size of 500 samples ...................... 83
    B-2-2 Rolling window size of 2000 samples .................... 84
## Table of Contents

### C Results

C-1 Normal operations dataset ................................................................. 86  
  C-1-1 Identified parameters using the OLS and ridge regression methods .... 86  
  C-1-2 Results of regressions with concatenated variables ..................... 88  
C-2 Abnormal operations dataset ............................................................... 89  
  C-2-1 Amount of included predictors by the LASSO ............................... 89  
  C-2-2 Identified parameters using the OLS and ridge regression methods .... 90  
  C-2-3 Results of regressions with concatenated variables ..................... 92  

### D Validation Results

D-1 Quality estimator datasets for the Deisopentanizer .......................... 94  
D-2 Results of regression methods of the Deisopentanizer ....................... 95  

### Bibliography

Bibliography .................................................................................................. 99  

### Glossary

Glossary ......................................................................................................... 101  
  List of Acronyms .................................................................................... 101  
  List of Symbols ....................................................................................... 102
Preface

This report is the result of ten months of research at Shell and is my Master of Science graduation thesis. This thesis started as an evaluation of the possibilities of exploiting information from historic industrial plant data. The current state of practice and research within Shell, the process industry, and the academic literature were evaluated. The results of this was that this research focusses on the application of statistical methods to process monitoring in industrial plants. It explores the possibilities of using regression methods to predict key variables within industrial processes and evaluates the generality of these methods, by applying them to historic datasets of multiple processes at the Shell Pernis refinery.

This research was possible because of the process data and process knowledge which was provided by Shell. Therefore I would like to thank Shell for the opportunity to work on such an innovative project. In particular I would like to thank my supervisors ing. P. Kwaspen & ir. J. Hofland from Shell for their assistance during the writing of this thesis and Edwin Hessels for his support in providing his knowledge of the considered processes. I would also like to thank my academic supervisors prof.dr. R. Babuška, dr. R. Ferrari & dr. D. Tax for their assistance during this research.

Delft, drs. D.W.B. Bender
27th June, 2016
Chapter 1

Introduction

Modern industrial plants are often large and highly complex installations. These include an enormous number of sensors, which measure a variety of both process variables e.g. pressure, temperature, flow, etc., and diagnostic data, which provide information about the condition of equipment, including that of the sensors themselves. To give a sense of how large these numbers can be, let us take Shell’s Pernis refinery, which is the largest refinery in Europe, and one of the largest in the world. Over 400,000 barrels of crude oil are produced here each day, and to make this possible, more than 50,000 sensors are used to monitor the different installations and processes within the refinery.

For almost all modern industrial plants, the data from the sensors is stored in some sort of Data Historian database. This means that for these plants, a vast amount of (historic) data is available. Currently, operations and maintenance only make use of a relatively small part of the info contained within this data, mostly limited by the fact that this information has to be extracted from the data manually. So a higher level of op intelligence and understanding of the data, aided by a shift towards automation of information extraction, would lead to an improvement in the overall performance of the plant.

Over the past decades technological advances have led to impressive gains in computational power. These advances, combined with the enormous increase in number of sensors within industrial plants and the resulting increase in the amount of available (historic) data, have made possible new and more advanced analytical methods. Use of such methods would make it possible to exploit more of the information contained in (historic) plant data than is currently the case, significantly improving the overall performance of the plant.

More advanced analytical methods are also vital for the development and realisation of normally unmanned installations (NUIs). An NUI is a type of facility that is designed to be operated remotely without the constant presence of personnel. Ideally, NUI design should yield a highly reliable performance yet require little maintenance, minimising production downtime and reducing the frequency of site visits. Shell’s chief technology officer, Yuri Sebregts, has stated that NUIs are to become the standard approach for many plants and production facilities.
One overview of the current state of process analytics, which covers both process and condition monitoring within industry, is given by the ARC Advisory Group (ARC) in [1], and is presented in Figure 1-1. They define four categories within process analytics: descriptive, diagnostic, predictive, and prescriptive. Methods that fall within the descriptive category are simple methods such as reports, that provide operators with only basic information about what has happened. This level of analytics does not provide the user with any means of interacting with the data or manipulating it; it simply provides a statement of facts (e.g. weekly production levels). Diagnostic methods help operators to understand simple cause-and-effect. The difference with the descriptive category, is that instead of merely seeing what has happened, it is also possible to understand why it happened. In practice there are a many more of methods which lie somewhere on the spectrum between descriptive and diagnostic. An example of a method that lies on the far right of that spectrum are visual data discovery tools. This type of tool allows a user to explore data, largely free of constraints. This can help users find answers to unanticipated questions [2]. Predictive analytics aims to identify hidden patterns within the data, using methods such as machine learning and other statistical methods. Prescriptive analytics goes one step further, suggesting the appropriate course of action that should be taken given a certain upcoming event.

Moving from descriptive and diagnostic towards predictive and prescriptive, means moving towards automation. In other words, there is a distinct shift towards using sophisticated algorithms to help make sense of the data. For the descriptive and diagnostic styles, software support is focused on fairly simple, but labour-intensive, data manipulation. However, extracting useful information from data is still very much dependent on the human insight. With the predictive and prescriptive approaches, that changes. Not only does the software perform mundane data manipulation, but it also takes the lion’s share of responsibility for identifying trends and patterns in the data.

For process operations to satisfy their performance specifications, closed-loop controllers are applied. Their goal is to compensate for effects of disturbances and changes in the system. However, the controllers are unable to deal with some of disturbances and changes in the
process which are therefore defined as faults. In order for a system to operate at the perfor-
mance specifications, these faults need to be detected, diagnosed and removed. These tasks
are associated with process monitoring. Yet process monitoring applications within Shell,
and the rest of the industry, have changed very little in the past decades. Although more
advanced methods have been developed and researched in the literature, their application to
real-world industrial plants has not yet become standard.

The use of machine learning techniques and other statistical methods is key to moving towards
the predictive and prescriptive levels of analytics. These are data-driven methods which are
able to extract patterns from data and within industrial plants. These patterns can be used
to construct multi-step ahead predictions which can predict both process and equipment
failures. If this approach can accurately predict key variables in the process, these predictions
can be used to check whether there is any indication of an problem. Achieving this would
increase process flexibility, process quality, asset health and incident prediction which would,
in turn, yield many benefits such as reduced unplanned downtime, waste, operating costs and
operational risk, as well as increased worker safety.

The amount of available process data within Shell makes it possible to perform research on
the use of predictive analytics for process monitoring purposes in real-world industrial plants.
This can be translated into the following research goal:

- “How can machine learning and other statistical techniques be implemented to make
earlier fault detection possible and, as a results, improve the performance of a plant?”

The research goal is evaluated by applying different regression methods to a large set of input
variables, and using these to construct predicted multi-step ahead values of key variables
within the process. The regression methods must be able to deal with the large amount of
input variables and must also be able to determine whether input variables contain relevant
information; if not, they need to be discarded. This research uses data from a crude distiller
unit at the Shell Pernis refinery as a case study.

Before the analysis and implementation of the regression methods can be done, two issues
need to be discussed. Furthermore, a decision must be made about how to deal with each
of them. The first issue deals with the sampling rates of variables in the case study. The
sampling rates of each of the input variables is once per minute. However the sampling rates
of the key variables are both irregular and more frequent than once per minute. The result is
that if a dataset of samples containing both input and output variables is made, part of the
input samples will be missing the corresponding output value. Therefore, a decision needs to
be made about whether to use some sort of estimation method to fill in these missing values,
or otherwise the data samples, for which the output value is missing, should be discarded.
The first option adds estimation uncertainty to the dataset and the second option discards
potentially valuable information contained in the input variables.

The second issue that should be evaluated, is how to model the dynamics of the process. For
a multi-step ahead prediction, a multiple one-step ahead prediction, which can lead to the
multi-step ahead prediction, could be constructed. However, it is also possible to construct
A multiple one-step ahead prediction directly without first having to construct a prediction
for the intermittent time steps.
Once both these issues have been evaluated and a decision about how to deal with them has been made, the different regression methods can be analysed and evaluated. As it is not known precisely what type of faults have occurred and when, historic data sets which are a good representation of both normal and abnormal operations should be chosen. The performance of the regression methods is evaluated on both types of dataset. If regression methods perform well on these datasets, the same methods are subsequently be applied to a different process, in order to validate their performance and robustness.

The rest of this report is structured as follows. First of all, the fundamentals of process monitoring are discussed in Chapter 2. Then in Chapter 3 the case study and its variables are discussed. Subsequently, in Chapter 4, the issues of the dual and irregular sampling rate and the choice of dynamic model are investigated. In Chapter 5, the different regression methods are discussed and in Chapter 6 an analysis of each method to the historic data is performed. The results of the application of the regression methods to the normal and ‘faulty’ historic datasets are presented and evaluated in Chapter 7. The validation of the best performing regression methods to a different process of the case study is done in Chapter 8 and the generated results are presented and discussed. The final chapter discusses the conclusions of the research, and presents its discussion and recommendations.
Chapter 2

Fundamentals

This chapter gives a brief description of process monitoring and associated definitions.

2-1 Process monitoring

For process operations to satisfy their performance specifications, closed-loop process controllers are applied. The process controllers that operate the plant can often be split into two layers, basic and advanced. The basic controllers are low level controllers which look at (small) subsets of variables that usually have fast dynamics. Their objective is to guide the variable to its reference value, the setpoint. Conversely, the goal of the higher advanced level controllers, which are often called supervisory level controllers, is to look at the entire system. They have slower dynamics, but optimise the entire system by assesses all the variables within simultaneously and altering their set points. Within Shell these two layers are referred to as Base Layer Control (BLC) and Advanced Process Control (APC) respectively.

Their goal is to compensate the effects of disturbances and changes in the process. However, within the process there are also a number of disturbances and changes in the process which the process controllers are not able to deal with; these are defined as faults. For process operations to satisfy their performance specifications, the faults within the process need to be detected, diagnosed and removed. These tasks are associated with process monitoring. Achieving early and accurate fault detection and diagnosis, can minimize downtime, increase safety, and reduce costs for plant operations. The objective of process monitoring can be defined as ensuring the success of the planned operation by recognising anomalies of the behaviour and by developing measures that are maximally sensitive and robust to all possible faults [3]. The goal of the early fault detection and diagnosis is to have enough time for counteractions such as other operations, reconfiguration, planned maintenance or repair [4].

The general idea of process monitoring can be described as follows; first a fault detection method is applied to data from the process which generates features which describe the current state of the process. Subsequently these features are compared to a threshold or subjected to a different change detection method, which assesses whether the process is still
in its normal operating state. If this comparison concludes that the current state of the process differs from its normal operating state, a symptom is generated. It is possible that multiple features and/or multiple symptoms are used. Finally a fault diagnosis method is applied to the generated symptoms, to determine what type fault has occurred. This procedure is illustrated by Figure 2-1, where \( x \) is the data from the process, \( f \) the features generated by the fault detection method, \( s \) are the symptoms, and \( F \) are the determined faults.

![Figure 2-1: General scheme for process monitoring](image_url)

### 2-1-1 Process monitoring definitions

Throughout the literature on process monitoring a number of different taxonomies are used. In this report the definitions associated with process monitoring are adopted, which are given in [4]. First of all the definition of a fault is given;

"A fault is an unpermitted deviation of at least one characteristic property (feature) of the system from the acceptable, usual, standard condition."

A fault may lead to either a failure or malfunction. These two terms are defined as:

- "A failure is a permanent interruption of a system’s ability to perform a required function under specified operating conditions"  
- "A malfunction is an intermittent irregularity in the fulfilment of a system’s desired function"

Both a failure and a malfunction can be caused by a single or multiple faults. The possible chain of events are illustrated by Figure 2-2.

Within process monitoring two parts can be identified. These are fault detection and fault diagnosis [4]. These are defined as:

- **Fault detection**: determining that a fault has occurred.
- **Fault diagnosis**: determining which fault occurred, i.e. determining the cause of the observed out-of-control status. More specifically determining the type, location, magnitude, and time of the fault.

There exist a number of other terminologies throughout the literature, including one which further distinguishes the fault diagnosis part into fault isolation and fault identification, here the first isolates which kind of fault is present, and the latter further characterises it. As an
example let us take a tank in which a leak can occur. The process monitoring sequence for this terminology would be, first the fault detection system shows that something is wrong with the tank. Second the fault isolation determines that it must be a leak. And finally the fault identification determines the size and location of the leak. As specified, in this research we group the fault isolation and identification together into fault diagnosis.

2-1-2 Fault detection methods

Within the research area of fault detection a number of divisions are described for the different methods. In this report the methods of fault detection are split into two main categories; data-driven methods and model-based methods.

Data-driven fault detection methods use only (historic) process data to detect fault within the process and require no prior knowledge about the process. These methods are trained with non-faulty data, and are used to recognise deviations from this normal behaviour, which are caused by faults. The most simple example of this is limit-checking. This method analyses historic data of a single variable and determines what its normal bounds are. These bounds are typically a maximum, upper control limit (UCL), and a minimum, lower control limit (LCL). If the value of the process variable crosses one of these bounds, a fault is declared. This is illustrated in Figure 2-3a. However checking just one variable is often not sufficient to properly detect faults. Process variables are often correlated and this means that one process variable can have a significant influence on the acceptable bounds of another variable. Applying multivariate limit-checking allows one to determine the bounds of acceptable operation for several process variables simultaneously. This is illustrated by Figure 2-3b.

It is also possible to apply some sort of statistical transformation, such as principal components analysis (PCA) or partial least squares (PLS), to the process variables to reduce the dimensionality. In this case limit-checking can be applied to one or more of the transformed variables. Limit-checking is one method that can be used to detect deviations from normal behaviour, but there are many other methods, such as change detection, that can also be applied. Examples of this are deviations in mean, variance or stationarity. The main drawback of data-driven methods is that its performance is highly dependent on the quantity and quality of the process data.
Model-based fault detection methods make models of a process, and compare their outputs to the outputs of the real process. This is illustrated in Figure 2-4. In the figure the measured input signals $U$ and output signals $Y$ are shown, and their relationship is represented by a mathematical process model. The corresponding fault detection method extracts features, such as parameters $\theta$, state variables $x$ or residuals $r$. By applying a change detection method to compare these features to the normal behaviour of the system, analytical symptoms $s$ are generated. The noise in the system is represented by $N$.

Many implementations of model-based fault detection methods use the residual between the output of the real process and the output of the process model as the feature which they use for fault detection. The general procedure of this is as follows; a model of the process is constructed, and this model is used, at some point in time $\tau$, to generate an estimate of the output variable $\hat{y}_\tau$ using the input variables from the previous time step, $x_{\tau-1}$. The residual
can then be used as the feature for fault detection. It is given by:

\[ r_\tau = y_\tau - \hat{y}_\tau \]  

(2-1)

The idea here is that the model represents the process under normal non-faulty conditions, and that its output represents the output of the system if no fault was present. It is thus assumed that if no fault is present in the system, the residual should be zero. However if a fault occurs in the system, the output of the process is affected by the fault, while the output of the model is not affected. In this case the residual will be non-zero, and it can be concluded that a fault has occurred. In practice, noise will also cause the residual to be non-zero even if there is no fault in the system, therefore a threshold is introduced \( \bar{r}_\tau \). If the residual crosses this threshold, a fault is detected. This procedure is illustrated by Figure 2-5.

![Figure 2-5: Example fault detection method](image)

There are many different methods that can be used to construct a process model. If the structure and the parameters of the process are known, first principles can be used. In this case the model is constructed using the physical properties of the system, which are governed by the laws of nature. Applying this method to large systems is done by starting to model each of the subsystems and subsequently combining them into a single overall model of the system. This kind of modelling is called theoretical modelling and it always starts by making assumptions about the process, which simplify the construction of the model. In large real-world applications, theoretical modelling is not always possible in practice. The system could be so complex that constructing its model would either cost too much effort or could even be too complex to model. In these cases experimental modelling can be applied. This is often called identification, and this type of modelling obtains the mathematical model of the process by using measurements. The methods use measurements of the input and output signals, which it evaluates in such a way that their relation is expressed in a mathematical model. Techniques such as linear regressions or artificial neural networks are examples of this. If such methods are used, it means that data-driven techniques are used to construct a process model; this differs from the data-driven fault detection techniques which they simply use data-driven techniques to analyse the process data.

The theoretical models are built on a functional description of the physical data of the process and its parameters. The experimental model, on the other hand, determines its parameters from measurements, whose relation to the physical data and process is unknown. Therefore the latter are referred to as a black-box models. In contrast, the theoretical models are referred to as white-box models. However, it is not always the case that a model can be classified as either black- or white box. For instance, when a model knows the physical laws, but doesn’t know the parameters, and the process measurements can be used to identify these. Such models are referred to as grey-box models [4].
2-1-3 Fault diagnosis

Fault detection methods are used to generate features and these features are subsequently subjected to some sort of change detection, which compares the current state of the process to its normal behaviour. The results of this is that it is possible to detect whether a fault has occurred, and additionally a set of symptoms for this fault can be generated. The next step within process monitoring is to determine which type of fault has occurred. To achieve this fault diagnosis is applied. After the symptoms are generated, a fault diagnosis method is needed to determine what type of fault has actually occurred. Within most industrial plants, it is currently the operator of the process that performs the fault diagnosis. The operator assesses the symptoms of generated by the process and fault detection methods, and determines: what is going on, and subsequently, what can be done to correct this?

More specifically, the task of diagnosis is to separate $n_f$ different faults, $F_j, j \in \{1, \ldots, n_f\}$, using $n_s$ symptoms $s_i, i \in \{1, \ldots, n_s\}$. The faults are combined into a fault vector $\mathbf{F} = [F_1 \ F_2 \ \ldots \ F_{n_f}]$ and the symptoms into a symptom vector $\mathbf{s} = [s_1 \ s_2 \ \ldots \ s_{n_s}]$. Most of the methods compute a fault measure $f_i$ for each fault class $F_j$. The most probable fault is then given by the fault corresponding to the maximum value of $f_j$. However, in reality, not only the largest $f_j$ is of relevance; unclear situations and measurement noise can create high values of multiple fault measures. This can indicate uncertain decisions. Hence, all values of $f_j$ are important when determining the diagnosis of a system.

There are two main methods of fault diagnosis; classification methods and inference methods. If no information is available on the fault-symptom causalities, experimentally trained classification methods can be applied for fault diagnosis. This leads to an unstructured knowledge base. If the fault symptom causalities can be expressed in the form of if-then rules then reasoning or inference methods are applicable. Within these two main methods, there are a number of different fault diagnosis methods which can be applied.

Automating the fault diagnosis in an industrial plant would require at the least, either structural knowledge between possible symptoms and faults within a process or a sample set of data, which contains enough occurrences of each type of fault. In a modern industrial plant, the process are often too complex for the first option to be possible. The second option might seem like the more attractive option, as there is a vast amount of plant data available, however operators almost never let a plant run for the full length of a fault, but rather compensate for it or even shut down the process. Therefore there are no data sets available, that contain enough fault occurrences to make automated fault diagnosis possible.

2-1-4 Early fault detection

In this research we are interested in whether large sets of data can be used to make earlier fault detection possible. Most of the model-based fault detection methods aim to detect faults through their effect on monitored variables. They apply some sort of change detection to a feature which was generated with the estimate of a monitored variable, e.g. a residual constructed with the estimated output of a process and the real output. One possibility to achieve earlier fault detection is to make multi-step ahead predictions about these monitored variables. To achieve this we continuously learn a model, which after the fault time will include the fault as well, and use it to predict the system behaviour a number of steps ahead.
These predictions can then be compared to the normal system behaviour, and if a change is detected, it can be concluded that a fault has occurred. The general scheme for this is illustrated by Figure 2-6. Here $x_t$ is the data from the process at time $t$, $\hat{f}_{t+h}$ is an estimate of the features generated by the fault detection method at time $t + h$, $\hat{s}_{t+h}$ is an estimate of the symptoms at time $t + h$, and $\hat{F}_{t+h}$ is an estimate of the determined faults at time $t + h$. For the construction of the prediction model, which is necessary for the early fault detection, machine learning or other statistical methods are evaluated.

The most straightforward way to use predictions of system behaviour for early fault detection is limit checking. In this case a prediction model is used to construct prediction about some output variable $y$ from input variables $x$. Subsequently limit checking can be applied to the predictions $\hat{y}$, and if it is determined that a prediction crosses one of the limit values, a symptom is generated. This method is illustrated by Figure 2-7. It is of course also possible to apply some other change detection method than limit checking, but the idea remains the same.

Another, slightly more complicated, method for achieving earlier fault detection, is predicting both the current system behaviour and normal system behaviour. The current system behaviour could be either normal or fault, depending on whether a fault has occurred. A residual is constructed between both predictions and this residual can be evaluated to assess whether a fault has occurred. This method can be seen as a predicted version of parity equations and can be described by:

$$\hat{r}_{t+h} = \hat{y}_{t+h}^{normal} - \hat{y}_{t+h}$$  \hspace{1cm} (2-2)\

where $\hat{y}_{t+h}^{normal}$ is the prediction of $y_{t+h}$ under normal conditions, and $\hat{y}_{t+h}$ the equivalent prediction under current conditions. The entire scheme is illustrated by Figure 2-8. The difficulty with this method is that, apart from having to construct an accurate forecast model, it is also necessary to construct a model that accurately predicts the output of the
Figure 2-8: Early fault detection scheme with predicted parity equations

system, if the system were in its normal operating state. Additionally, any model or measurement uncertainty will propagate over time, making predictions on a long time horizon useless.

One idea is that the same identification procedure, such as a certain neural network or linear regression method, can be used to determine both the normal situation forecast model and the forecast model. The first can be constructed using a set of measurements that are obtained during a period of ‘normal’ operations. The latter can be constructed using the the most recent measurements, i.e. for time $\tau$ use $t = \tau - \tau w - h, ..., \tau - h$. As this model would be identified using normal operating data, it should not take any effect from a fault into account and construct a forecast of the output variable in the case that no fault was present in the system.

Another option would be to use a physical model, which gives an estimate of a future value, and use this estimate as the forecast for the normal situation. As it is based on the physical properties of the system it should also not take any effect from faults in the system into account. If these methods are compared to the classic parity equations method, the physical or normal situation prediction model could be compared to the process model, and the forecast model acts to the real process of the system, as the latter take all effects from faults into account, whereas the physical model and process model do not.

In both cases the effects of model and measurement uncertainties will be present, and increase when the time horizon becomes long. It should therefore be investigated whether these types of fault detection methods would be more effective than those where the predicted output is compared to a simple limit value. Analysing the variances of the prediction models could help with the investigation.

Examining the literature on process monitoring, which focus on early fault detection, it can also be concluded that real-world implementations are very limited. Most of the papers use some sort of simulator, where it is possible to keep the process in the normal operating state, and being able to know exactly when a fault is introduced, and what type of fault it is.
In this research a dataset from a real-world industrial plant is introduced. There are a number of requirements that it has to adhere to; the amount of data, the frequency of the data and the amount of filtering that has been applied to the data. The Shell refinery at Pernis agreed to provide the data from their Crude Distiller X1 (CDX1) unit for the case study of this research. As a result of a previous project, all the historic data from the unit for the whole of 2015 is available at sampling frequency of one measurement per minute. The dataset therefore met all the predetermined requirements. In the rest of the chapter, the CDX1 will be discussed in more detail. One of its subsystems, the Depentanizer (C-X11), is highlighted and its available data is analysed.

3-1 Crude Distiller X1

At Shell refinery at Pernis there are around 60 different factories that produce petroleum products and petrochemicals from crude oil. In total 20 million metric tonnes per year of crude oil is processed, this is over 400,000 barrels of oil a day. Two of the factories are the two equivalent crude distiller unit (CDU)s, the CDX1 and the Crude Distiller X2 (CDX2). All the crude oil that arrives at Shell refinery at Pernis must first go through one of these two units before it can be further processed.

The goal of the CDUs is to separate the crude oil into a range of petroleum products. The separation is achieved by using a number of subsequent distillation columns. Distillation columns split an incoming product into a top- and a bottom flow, where the boiling range of the different compounds and the temperature in the distillation column determines the composition of the two flows. The CDUs separate the crude oil into products such as: LPG, gasoline, naphtha, kerosene, diesel oil, motor oil and long residue.

In this research the data from the CDX1 is taken. A schematic overview of the unit is given in Figure 3-1, where the blue units represent the distillation columns.
From the CDX1 a single subsystem, the C-X11, is chosen as the initial case study of the research. Once different models have been identified, tested and evaluated on the C-X11, these models can be applied to other subsystems as validation and robustness analysis.

A short discussion of the other eligible subsystems is given. The set-up of the Deisopentanizer (C-X12) and Deisohexanizer (C-X13) resemble that of the C-X11. However the boiling points of their top- and bottom flow are much closer together than that of the C-X11. Therefore they are both considered superfractionator distillation columns. To achieve the separation the settling time is much longer, in the range of half a day, as opposed to an hour and a half for the C-X11. The first distillation columns in the CDX1 are the Crude Splitter 1 (C-X21) Crude Splitter 2 (C-X22). Both systems are a more complex version of the C-X11, i.e. there are around 100 process variables that can be used as input variables for the models, where the C-X11 only has around 35. Finally the Debutanizer (C-X27), Gasoline Splitter (C-X28) are both similar systems to the C-X11, but both processes are faster, which means they have a shorter settling time.

### 3-2 Depentanizer

The initial research is performed on a single subsystem of the CDX1, the Depentanizer (C-X11). This unit is fed the top product from the C-X28, and it splits this into the products pentane (C₅), both n-pentane (nC₅) and isopentane (iC₅), and heavier compounds. In the C-X11 subsystem, there are 35 process variables which are measured by sensors and whose data is stored in the data historian. A schematic overview of the subsystem is given in Figure 3-2.
3-2-1 Quality variable QTXX1

The analyser QTXX1 (QT1) measures the quality of the distillation column C-X11. This analyser is a chromatography detector, which measures the mass fractions of the different components in the bottom product. The quality of the C-X11 is defined as the mass fraction of nC5 in its bottom product. The analyser has an measurement error of 0-0.075% nC5, which is about 0-2% of the output value. The mass fraction QT1 is taken as the output of the system.

To determine the mass fractions of the different components, the chromatography detector takes a sample of the bottom product and analyses this sample. The analyser, and all of the other sensors, send their measured values to the distributed control system (DCS). The DCS performs the Advanced Process Control (APC) and also sends the measured data to the data historian, who stores the historic data. However, the analysis of the chromatography detector takes some time to be performed. The analyser typically outputs a value for QT1 every five or six minutes. In addition, in some cases the analyser rejects the result of a sample, and does not output to the DCS.

The result of this is that the QT1 has an irregular sampling time. The DCS uses a zero-order hold filter, which means that the last value of QT1 is kept until a new value is available. An added complication is that noise is added between the analyser and the DCS, as the analyses is connected through an analogue output, which can cause deviations in the output. These deviations are can be considered negligible, as they are in the order of $10^{-4}$. However, these deviations should not be considered new output values of the analyser. This can be done by implementing a deadband for the deviation of QT1. If the deviation of QT1 is smaller than the deadband, its value remains the same. In Figure 3-3 an historic dataset for the QT1 is shown. The blue line describes the dataset which is stored in the data historian and the
red dots represent the dataset, where only the new measured values from the analyser are taken. To generate this second dataset a deadband of 0.01 has been applied to eliminate the deviations caused by noise.

![Figure 3-3: Output dataset stored in data historian and the measured output dataset](image)

All the other variables in the C-X11 have a sampling time of once a minute. The goal is to construct a model using these other variables as inputs and QT1 as the output. This means that a so-called dual sampling rate problem occurs. The inputs of the model have a sampling rate of one per minute, and the output of the model has a sampling rate which is slower, and also irregular. The steady state of the C-X11 is reached after roughly 75 min. The analyser outputs a value of QT1 every 5 or 6 minutes, with an occasional exception where it takes about twice that time. The irregular sampling time is thus still shorter than the steady state time. This could mean that not a lot of information is lost, and that enough of the information is left such that the models are able to capture the dynamics.

Several approaches can be taken to deal with the dual sampling rate problem. The first is list-wise deletion, where every measurement that does not contain a complete set of all variables is deleted from the dataset [5]. In our case this would mean only using the measurements for which the analyser has given an output of QT1, and the corresponding input variables at that time. As a result all the measurements of the input variables, for which there is not an output value, would be deleted. The advantage of this method is that only measurements are used that are a direct output of the sensors, but this also means that all the information contained in the deleted input variables is not made used of. A second option is using interpolation between the measured values of QT1, to construct a complete dataset. Interpolation can be done using a number of different methods, such as linear, zero-order hold, or spline. The advantage of interpolation is that a complete dataset is available, and all the information contained in the input variables is utilized. However [5] warn, that one should be careful when using interpolation, especially when the time between known measurements is large. A third method to deal with the dual sampling rate problem is to apply a method, such as expectation maximization (EM), to estimate the missing values of QT1. However as the goal of this research is to identify a model using this data, this would mean that a model would be identified using estimated data, and a double estimation error would occur. This is undesirable, so estimation errors will not be examined in this research. Both list-wise deletion and three interpolation methods are evaluated in Chapter 4.
In Figure 3-4 the historic dataset of QT1 is presented for the entire year 2015 and first months of 2016. It can first be observed throughout the year 2015, there are several relatively volatile periods, such as the first four months of 2015, the months September and October in 2015, and the end of February through the beginning of March 2016. The periods June through July, November through January, and the end of March show less volatile behaviour. One consideration is that this data has not been filtered for periods of time where operation was shut down or other undesirable behaviour occurred. Another consideration is that throughout the entire year a project was ongoing to implement a new APC on the C-X11. The goal of the APC is to keep QT1 on specification, while minimizing the required energy from the reboiler. Ideally the QT1 would be as close as possible to zero, as the goal of the depentanizer is to send all the normal-C5 in the top product of the distillation column. It can be assumed that the new APC is responsible for most of the decrease in volatility, and that this APC will continue to be implemented on the C-X11. Therefore it can also be assumed that the data from the second half of the dataset, where the new APC was active, is more representative of future behaviour, than the data from the first half.

For the initial analysis a dataset is required where no fault occurs and where the distillation column operates at a given setpoint. For the process both alarm logs from the control system and shift logs from the operators are available. However these logs are not always properly or consistently updated. This means that there is no clear log available of when the process operated at normal conditions or when it operated at abnormal conditions. In addition, when the process was operating at abnormal conditions, it is not necessarily clear, what the cause was, as this could be induced by operators or any number of faults, that could occur in the process. To deal with this issue, a subset of data is chosen which approximates normal operating conditions. This subset is chosen by identifying a significant period of time, where the output remains at steady state. The data from the period spanning December 10th through December 19th 2015 is chosen for this. The data from this period of time is henceforth referred to as normal operations dataset (NOD) and it illustrated by Figure 3-4b. After analysing a dataset which approximates normal operations, analysis is performed on a dataset that includes some sort of fault. Therefore a dataset is identified which includes both an approximation of normal behaviour and a sudden fault entering the process. For this purpose the dataset the period spanning January 1st through January 17th 2016 is chosen. The data for this period of time is henceforth referred to as abnormal operations dataset (AOD) and is illustrated by Figure 3-4c. It can be seen, that at the end of the 11th of January, some sort of deviation enters the process and the output variable begins to deviate from its normal operating state. The process then recovers at the end of the 12th, but a few hours later it begins to deviate again.

3-2-2 Quality estimator QEXX1

In Shell’s control systems, a software package called Robust Quality Estimator Pro (RQE) is used to calculate process values, which can be used for closed loop control purposes. Within the C-X11, RQE is used to construct an estimate of QT1. This estimate is called QEXX1 (QE1). The values of QE1 are estimated using four input variables which were identified using the physical characteristics of distillation columns. These input variables are: TTX01, PTXX1, XCCX1, and FCXX3. An overview and description of the discussed variables is given in Table 3-1. In Appendix A-1 an overview and description is given of the entire set of
**Figure 3-4:** Entire dataset of QT1 and two subsets of its data; normal operations dataset (NOD) and abnormal operations dataset (AOD)
input variables that are available in the C-X11.

<table>
<thead>
<tr>
<th>Full Name</th>
<th>Name</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDX1:QTXX1.PV</td>
<td>QT1</td>
<td>wt%</td>
<td>Mass fraction nC5 in bottom C-X11</td>
</tr>
<tr>
<td>CDX1:QEXX1.PV</td>
<td>QE1</td>
<td>wt%</td>
<td>Estimate mass fraction nC5 in bottom C-X11</td>
</tr>
<tr>
<td>CDX1:TTX01.PV</td>
<td>TTX01</td>
<td>℃</td>
<td>Temperature at tray 11 of the C-X11</td>
</tr>
<tr>
<td>CDX1:PTXX1.PV</td>
<td>PTXX1</td>
<td>barg</td>
<td>Pressure at tray 11 of the C-X11</td>
</tr>
<tr>
<td>CDX1:XCXX1.PV</td>
<td>XCXX1</td>
<td>MW</td>
<td>Energy transfer from C-X21 to E-Z72</td>
</tr>
<tr>
<td>CDX1:FCXX3.PV</td>
<td>FCXX3</td>
<td>MT/day</td>
<td>Input flow of C-X11</td>
</tr>
</tbody>
</table>

The estimation of QE1 is done by first taking a linear combination of the four input variables to construct the linear output QE1\textit{lin}:

\[ \text{QE1}_{\text{lin}} = \hat{\theta}(1) \cdot TTX01 + \hat{\theta}(2) \cdot PTXX1 + \hat{\theta}(3) \cdot \frac{1000 \cdot XCXX1}{FCXX3} + \hat{\theta}(4) \]  

(3-1)

In a distillation column there are two important properties, that can be used to describe its performance. These are the effective cut point (ECP) and the separation index (SI). Cut points, within distillation operations, are key parameters. They are the temperatures at which the various distilling products are separated. Each product has two cut points; these are the temperature at which a product begins to boil, the initial boiling point (IBP) and the temperature at which it is 100% vaporized, the end point (EP). The ECPs are the point at which the products can be considered effectively clean. The SI gives an idea on how difficult the separation is to perform. For distillation this is the relative volatility of the two key components. When the relative volatility is one, there is no separation of components when distillation is used. In (3-1), the TTX01 and PTXX1 represent the ECP, and the XCXX1 and FCXX3 represent the SI.

The parameters \( \theta(1), \theta(2), \theta(3) \) in (3-1) are estimated off-line and are given by (3-2). The initial value of the bias term \( \theta(4) \), is also estimated off-line. However, the bias term is updated on-line, by correcting for the error between QT1 and QE1.

\[ \hat{\theta} = [ -0.4 \ 7.35 \ -1 \ \text{bias} ] \]  

(3-2)

When constructing the linear relation between input variables and QE1, the process control engineers chose to only use these four input variables, and not investigate whether adding extra input variables could increase the accuracy of the estimate. The reason for this is that they are able to justify using these four variables, because of the known relation between the quality of a distillation column and the ECP and SI.

If an extra input variable was found to increase the model fit and therefore added to the model, it is possible that the accuracy of the model could increase while the process conditions were normal. However if the conditions would deviate too far from the normal situation, the added variable might could possibly cause unwanted and unexpected behaviour in the estimate. As the estimate is used in the closed-loop control, this could lead to dangerous situations. This concept is called overfitting.
A reason that this can occur for this can be that the dataset that is being used for the training contains not only information about the regularities in the mapping from input to output, but also a sampling error. This means that there will be accidental regularities because of the fact that that particular training data was chosen. If this data is used to train the model, it won’t be able to tell which regularities are real and which are caused by the sampling error. As a result it will fit both kinds of regularity and if the model is flexible enough it will be able to model the sampling error really well. If this happens, the model will fit the training data better, but the performance will decrease when applied to new data.

To further illustrate the concept of overfitting, let us take the extreme case where there is a linear regressor, with as the same amount of parameters as number of samples in the training data set. In this case it is possible to, trivially, tune the parameters to exactly reproduce the samples, and it is very clear that this cannot be called learning. An example of overfitting is given in Figure 3-5.

This consideration should also be taken into account throughout the rest of this research. However the focus is first to assess whether using larger input variable datasets can produce accurate estimates, and the analysis of their robustness to operating conditions can be done as a later step.

To account for the non-linearity in the system, subsequently a piecewise linear function is used to transform the value of $Q_{E1lin}$ into $Q_{E1}$. This function is illustrated by Figure 3-6. A model of this type, i.e. a linear model followed by a static non-linearity is a Wiener non-linear model.

In Figure 3-7 a dataset of $Q_{T1}$ is given and the corresponding values of $Q_{E1}$, which was constructed using given linear relation and the piecewise linear function transformation. It can be observed that $Q_{E1}$ is able to capture a large part of the dynamics in its estimation of $Q_{T1}$, even before the analyser relays the value of $Q_{T1}$.

The fact that the given model is able to construct an estimate of $Q_{T1}$ from four input variables, that captures most of the dynamics of $Q_{T1}$ ahead of the analyser, means there is clearly a relation between $Q_{T1}$ and these four input variables. Therefore we will first do some analysis in Chapter 4 by using the output and just these four input variables, before we move on to using all the available inputs.
Figure 3-6: The piecewise linear function

Figure 3-7: Dataset of QT1 and QE1
3-2-3 Process variables TTX01, PTXX1, XCXX1, and FCXX3

The four input variables, which are used for the construction of QE1, are examined further. There four variables together are henceforth referred to as 4VAR. The NODs of each of the input variables are shown in Figure 3-8. As mentioned previously, the input variables are measured each minute, contrary to the output variable. There seems to be quite some correlation between TTX01 and PTXX1. In addition XCXX1 is relatively noisy compared to the other inputs. It can be observed that each of the four variables change quite a bit throughout the period, whereas we know that the output remains fairly constant in the same period of time.

![Figure 3-8: Datasets of the 4VAR variables](image-url)
Chapter 4

Model analysis

The goal of this research is to determine whether a model can be identified which is able to produce accurate robust predictions of the output of the system, without using prior knowledge about it.

For the identification of the model statistical learning techniques are evaluated. Statistical learning is a field whose objective it is to use statistical methods to learn from data. In general a set of input variables, which are often referred to as predictors or regressors, are used to construct predictions about one or more output variables, which are often called features [6]. To achieve this a training set of data, which includes both the outputs and the inputs, is required. This data is used to construct a prediction model, which uses new measurements of the inputs $x_t$ at time $t$ to make a predictions $\hat{y}_{t+h}$ about the outputs $y_{t+h}$ of the system at time $t+h$. This is illustrated by Figure 4-1.

![Figure 4-1: Prediction model example](image)

The problem that has been described is a supervised learning problem. This refers to the fact the the training set consists of both inputs and outputs, hence the goal is to learn a method to map the inputs to the outputs. If the outputs are not known, the problem is defined as a unsupervised learning problem. The goal in this case is often to find hidden patterns, which can be used to describe how the data is organized. A third option is reinforcement learning. In this case the model interacts with a dynamic environment, in which it must perform a given goal. However the model will not be explicitly told whether it has come close to its goal.

The output of a supervised learning problem is usually either quantitative or categorical. In the first case the output is a numerical value and this is referred to as a regression problem. In the latter case, the output is a state, such as on/off. This is referred to as a classification problem. Within this project the assumption is made that the outputs are quantitative and
available. Therefore this report focusses on the regression problem.

In this report the assumption is made that the processes can be described as *time-varying linear autonomous systems*. *Linear* refers to the fact that the output variable of the process can be described by a linear combination of input, autoregressive, and moving average terms. Another assumption is that the process behaves autonomously, which means that the inputs of the system are fixed and unknown. Finally *time-varying* refers to the fact that the parameters of the system change over time. To deal with this, it means that a model must be identified recurrently for each time step.

Before the modelling of the processes can be done, there are two questions which must first be answered. The first is what type of dynamic model should be used to model the processes. The second question is how the issue of the irregular sampling time and missing data of the output variables should be dealt with.

To evaluate what type of dynamic model should be used, two different models are tested, the finite impulse response (FIR) and autoregressive-moving average model with exogenous inputs (ARMAX). Their performance and model complexity are evaluated and a decision is made, which to use throughout the rest of this report. The evaluation is performed, by using both models to create predictions about QTXX1 (QT1), and comparing their performances. The irregular sampling time and missing data of the output variables can be dealt with in several ways. It is decided not to evaluate using complex methods for estimating the missing data, as this can lead to large estimation errors. Two, relatively simple, methods to deal with the issue are examined. The first is discarding all samples, which include missing data, this results in using a dataset which is smaller than the original dataset, but only contain measured data. The second method is using interpolation methods to estimate the missing data. The advantage of this is that, the interpolation methods are straightforward and no information is discarded. The separate interpolation methods are compared to using only measured data and a decision is made, which method is implemented in the rest of this report. The evaluation is performed by using ordinary least squares (OLS) to construct predictions of QT1, using each of the four methods.

The evaluation of the models and methods is done using only the 4VAR as input variables. The advantage this has, is that it is known that there is a relation between these predictors and QT1. With this relation, the best solution for the irregular sampling time can be found and choice of dynamic model can be made. Once these issues have been resolved, the research moves on to include a larger set of predictors, than just these four.

### 4.1 Performance evaluation

If different predicted datasets are constructed, their performance must be evaluated using some criterion to determine which of the models provides the most accurate predictions. Most criterion are based on the error function between the predictions and the realized output set at each sample throughout the datasets. However in the case of QT1, predictions can be constructed for each measurement, but due to the sampling time of the analyser, not each of these will have a corresponding measured output to be compared to. To solve this, the error function will still be used for the evaluation, but the error function will only be determined for
measurements for which a measured output is available. This means that not all predictions in
the output set will be used for the evaluation of the corresponding model. This is illustrated
by Figure 4-2. The black dotted lines represent the errors between the predicted data set
and the measured output set, and only these measurements are used for the evaluation of the
performance of the model.

4-1-1 Mean squared prediction error

To assess the performance of the different prediction models a measure for the accuracy is
required. The mean squared prediction error (MSPE) is proposed and is defined as:

$$\text{MSPE} = \frac{1}{T} \sum_{t=1}^{T} (\text{QT1}_t - \hat{\text{QT1}}_t)^2$$  (4-1)

It is not possible to say anything about the performance of the model by looking solely at
the MSPE as the value of the MSPE can differ greatly for models constructed from different
datasets of the same variable or amongst different variables. To address this problem, the
MSPE is normalised. This normalisation is done using the MSPE$_{\text{norm}}$ of historical mean
output set. The estimates in this output sets are determined as the average of the output
variables over the past 300 samples of the real output set. As the output variables have an
irregular sampling rate, this means that each estimate is not necessarily determined as the
average of 300 values, but as the average of the number of measurements that occur in 300
samples. This is illustrated by Figure 4-3. Here the dashed lines indicate the rolling window
of 300 samples and the red dots are the measured values of QT1 in this period of time. The
estimate for the output one time step after the end of the rolling window is the taken as the
average of these values and is indicated by the blue marker. After the estimate is generated,
the rolling window is shifted ahead one sample and the process is repeated.

The MSPE of the estimated dataset using the historical mean is denoted as MSPE$_{\text{hmean}}$. 
Then the MSPE of the prediction methods, \( \text{MSPE}_{\text{model}} \), are normalised as follows:

\[
\text{MSPE}_{\text{norm}} = \frac{\text{MSPE}_{\text{model}}}{\text{MSPE}_{\text{hmean}}} \tag{4-2}
\]

### 4.2 Dynamic model

One of the most general models that is used for time-varying linear autonomous systems, such as presented in Figure 4-4, is the ARMAX model, which is defined as:

\[
y_t = c + \sum_{i=1}^{p} \phi_i y_{t-i} + \sum_{j=1}^{q} \theta_j \varepsilon_{t-j} + \sum_{k=1}^{b} \beta_k x_{t-k} + \varepsilon_t \tag{4-3}
\]

where \( y_t, \varepsilon_t, x_t \) are respectively the output, the random error, and the exogenous variables at time \( t \). \( \phi_i, \theta_j, \) and \( \beta_k \) are model parameters for \( i = 1, \ldots, p \), \( j = 1, \ldots, q \), and \( k = 1, \ldots, b \), respectively. \( c \) is a constant. The simplified notation is given by:

\[
A(z)y(t) = B(z)x(t) + C(z)\varepsilon(t) \tag{4-4}
\]

where

\[
A(z) = (1 + \phi_1 z^{-1} + \ldots + \phi_p z^{-p}) \tag{4-5}
\]
\[
B(z) = (1 + \theta_1 z^{-1} + \ldots + \theta_q z^{-q}) \tag{4-6}
\]
\[
C(z) = (1 + \beta_1 z^{-1} + \ldots + \beta_b z^{-b}) \tag{4-7}
\]

and \( z^{-1} \) is the unit delay, i.e. \( z^{-1} x_t = x_{t-1} \).

Depending on the parameters \( p \) and \( q \), the ARMAX\((p,q)\) model takes past outputs and past errors into account, when identifying a model. An alternative for the ARMAX\((p,q)\) model is the FIR model. The FIR model is simply a restricted version of the ARMAX\((p,q)\) model, where the \( A(z) = 1 \) and \( C(z) = 1 \), which results in the following model:

\[
y(t) = B(z)u(t) + e(t) \tag{4-8}
\]
To evaluate the performance of both models an analysis is performed to verify whether using the restricted FIR model reduces the performance of the models in comparison to using the ARMAX(p,q) model. For the analysis the FIR model is tested against two ARMAX(p,q) models.

The analysis is performed by using each of these three models to construct a set of 20-step ahead prediction of QT1. The models are identified recurrently, at each time step, and each of these identified models is used to construct a single prediction. The models are identified using a rolling windows of 500 samples. The results of this analysis are given in Table 4-1.

<table>
<thead>
<tr>
<th>Model</th>
<th>MSPEnorm</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIR</td>
<td>0.42</td>
</tr>
<tr>
<td>ARMAX(1,1)</td>
<td>0.61</td>
</tr>
<tr>
<td>ARMAX(2,2)</td>
<td>0.63</td>
</tr>
</tbody>
</table>

It is clear that the results of the FIR model is better than the results of the two ARMAX(p,q) models. The MSPEs are around a third smaller. In Figure 4-5 the forecast set of the FIR and the ARMAX(1,1) model are plotted against the linearly interpolated real output.

It can be seen that the ARMAX(1,1) model lags behind the real output. The auto-regressive and moving-average part of the model can cause the outputs of the models to follow the real output, but it is clear that is cannot capture the dynamics of the systems. The FIR model
on the other hand, seems to be able to describe the dynamics of the process. It is therefore conclude that, within this report, the output of process is not dependent on past outputs or past errors and the system can be described using a FIR model.

4-3 Variable types and terminology

Before the irregular sampling rate is discussed, a brief description of some variable types and terminology, as well as a description of the procedure for using OLS for out-of-sample forecasting are given.

In this research we will restrict ourselves to a multiple inputs, single output (MISO) system. A single measurement of the output is defined by $y_t$ and a single measurement of input $i$ is defined as $x_{i,t}$, where $t$ denotes the $t$-th measurement in the dataset. The vector including all input variables at time $t$, is defined as $x_t$. A vector of measurements for input variable $i$ is referred to as $x_i$ and the matrix which includes a vector of measurements for multiple input variables is represented by $X$. For the vector of output variable measurements this is done equivalently as $y_t$.

In this research the focus is on the MISO regression problem. This problem can be loosely defined as follows; make a prediction of the output $y_t$, given the values of an input vector $x_{t-h}$. The prediction is defined as $\hat{y}_t$, where $h$ is the prediction horizon, which indicates how many the amount of steps ahead the prediction is. The construction of the prediction model is done with a training set. The training set is a set of measurements $V \equiv (X,y)$, where $X$ is a $N \times p$ matrix, in which $N$ is the number of measurements and $p$ the number of input variables, and $y$ is a $N \times 1$ vector.

4-4 Ordinary least squares

For the evaluation of the irregular sampling rate a FIR model and OLS is used to construct out-of-sample predictions of QT1. The procedure for this is as followed;

The linear regression model, with output $y_t$ and $p$ input variables $x_{t-h} = (x_{1,t-h}, x_{2,t-h}, \ldots, x_{p,t-h})$, is given by:

$$y_t = c + \sum_{j=1}^{p} \beta_j x_{j,t-h} + \varepsilon_t \quad (4-9)$$

$$= c + \beta x_{t-h} + \varepsilon_t \quad (4-10)$$

where $\beta$ are unknown parameters and $\varepsilon$ is the error. The term $c$ is the intercept, which is also referred to as the bias. In practice it is often convenient to include the constant variable 1 in $x_t$, i.e. include $c$ in the vector of coefficients $\beta$. In this case the model can be simplified to:

$$y_t = \beta x_{t-h} + \varepsilon_t \quad (4-11)$$

From here on it will be assumed that the $c$ is included in $\beta$ unless indicates differently.

D.W.B. Bender

CONFIDENTIAL

Master of Science Thesis
To fit the FIR model to the training set, OLS is applied. In this approach the coefficients $\beta$ are chosen, such that they minimize the residual sum of squares:

$$\text{RSS}(\beta) = (y - \beta X)^T (y - \beta X)$$  \hspace{1cm} (4-12)

The unique solution for the estimate of $\beta$, in this case, is given by:

$$\hat{\beta} = (X^T X)^{-1} X^T y$$  \hspace{1cm} (4-13)

In this research FIR is used to construct out-of-sample predictions. This means that if predictions are made about measurements $t + h$, only information is used which was available at measurement $t$. The implementation of this procedure is now described for the case where QT1 is taken as output variable and TTX01, PTXX1, XCXX1, and FCXX3 as the input variables.

Let us take a dataset of $N$ samples, containing five measurements for each time step, which correspond to the output and four input variables. At sample $t$ the output variable is denoted as QT1$_t$ and the vector of the four input variables as $x_t$. To construct forecasts using this method, first the FIR model is defined as:

$$\text{QT1}_{t+h} = \beta x_t + \epsilon_t$$  \hspace{1cm} (4-14)

here $h$ is prediction horizon. As a FIR model is used, only predictions about the output of the model at an instant in discrete time that is $h$ sample times in the future are made. Thus it is not the case that a prediction is made for each sample time along the entire discrete time interval spanning from $t$ to $t + h$.

Second, the vector of parameters $\beta$ needs to be estimated. For this estimation a training set of samples is necessary. The training set is determined by defining a rolling window of size $rw$. As the prediction horizon is larger than zero, new samples must be generated from the dataset, where each sample contains the input variables from time $t$ and the output variable at time $t + h$. For the first estimation of parameters $\beta$, the first $rw$ samples are of the dataset are taken. This means taking the first $rw$ measurements of the input variables $\{x_t\}_{t=1}^{rw}$ and the the coupled output variables $\{\text{QT1}_t\}_{t=1}^{rw+h}$.

The estimate of parameters $\beta$ is the linear least squares estimate of $\beta$, which is constructed by regressing $\{\text{QT1}_t\}_{t=1}^{rw+h}$ on $\{x_t\}_{t=1}^{rw}$. This estimate is denoted by $\hat{\beta}_{rw+h}$. For the the generation of this estimate the QT1$_{rw+h}$ has been used. As we want to use the most recent information available, the first out-of-sample forecast that can be constructed is the forecast using the input variables from the same instant in time $x_{rw+h}$. This forecast is the estimate of QT1$_{rw+2h}$, and is denoted as QT1$_{rw+2h}$. It is constructed using the estimate of $\beta$ as follows:

$$\hat{\text{QT1}}_{rw+2h} = \hat{\beta}_{rw+h} x_{rw+h}$$  \hspace{1cm} (4-15)

Subsequently, the second out-of-sample forecast is then constructed by:

$$\hat{\text{QT1}}_{rw+2h+1} = \hat{\beta}_{rw+h+1} x_{rw+h+1}$$  \hspace{1cm} (4-16)

where $\hat{\beta}_{rw+h+1}$ is estimated by regressing $\{\text{QT1}_t\}_{t=2}^{rw+h+1}$ on $\{x_t\}_{t=2}^{rw+1}$. Continuing in this manner throughout the whole dataset, results in a out-of-sample containing forecasts $\{\hat{\text{QT1}}_t\}_{t=rw+2h}^{N}$.
where $N$ is the number of observations. Technically the forecast set could continue until $N+h$, however the value of the output variable is not available in the dataset, and therefore these forecasts cannot be verified.

When constructing forecasts, using this procedure, there are two parameters which have to be defined. These are the prediction horizon $h$ and the size of the rolling window $rw$. It must be evaluated which settings both these parameters lead to the most desirable results.

### 4-4-1 Concatenated variables

It is also possible to concatenate the input variables. In this case the output is not just mapped from the input variables of one measurement, but using the variables of several measurements. For a $1 \times p$ vector of input variables $x_t$, where $p$ indicates the amount of predictors, this means replacing the vector $x_t$ by the $1 \times (p \times l)$ vector $[x_t \ x_{t-1} \ \ldots \ x_{t-l}]$, where $l$ indicates the amount included time steps. In this case the dynamical model of the process changes to:

$$QT_{t+h} = \beta [x_{t-h} \ x_{t-h-1} \ \ldots \ x_{t-h-l}] + \epsilon_t$$

(4-17)

If this approach is taken and the input variables are concatenated, we talk about using concatenated variables as inputs to the model.

### 4-5 Dual and irregular sampling rate

The identification of the process requires a set of corresponding input and output variables. The output variable set QT1 is missing measurements, and these can either be filled in using interpolation, such that the identification can be applied to a set of measurements which span the entire dataset. Alternatively the identification can be applied to only the available output measurements, and their corresponding input variables. To determine which of these options should be applied, each of them is evaluated in this section. Three different interpolation methods are evaluated; linear, zero-order hold, and spline. The performances of these three interpolation method are compared to the performance of using the measured set. These three methods are illustrated by Figure 4-6.

The evaluation is performed by using a FIR model and OLS, with QT1 as the output, and the 4VAR as input variables. The four different methods of generating the output set of QT1 are evaluated for six rolling windows, 100, 300, 500, 1000, 1500, and 2000, and prediction horizons ranging from 1 to 30. Additionally each of them is tested using 0 to 5 concatenated input variables. The results of these tests are presented in Appendix B-1.

The difference in results between the models identified with different numbers of concatenated variables is small in terms of MSPE. Therefore the analysis is continued with the models without concatenated variables. The results of the models with a rolling window of 500 and 1000 are presented in Figure 4-7.

It can be clearly seen that the performance of the models, that were identified only with the measured data sets, perform significantly worse than the models that were identified with interpolated data sets, as the MSPE is much worse.
Figure 4-6: Dataset of QT1 and output sets constructed with three different interpolation methods

Figure 4-7: Results of the OLS

(a) Rolling window of 500 samples

(b) Rolling window of 1000 samples
The differences between the three interpolation methods seem to be small. If a rolling window of 100 is used, the zero-order hold method performs slightly worse for lower prediction horizons. But if a rolling window of 1000 is used, the differences in MSPE are negligible. As the differences are small, it is chosen to use the linear interpolation method to complete the output set of QT1, as this interpolation method has shorter computation time than the spline method, and seems slightly more robust than the zero order hold method.

4-6 Comparison identified method and quality estimator

After deciding that a FIR model will be used to describe the dynamics of the Depentanizer (C-X11), and that a linear interpolation method will be implemented to estimate the missing output values, the resulting model is compared to QEXX1 (QE1).

Figure 4-8 shows the linearly interpolated QT1 outputs, the estimate QE1, and the forecast set constructed with the FIR model and the same four input variables as the QE1 uses. It can be seen that the FIR model shows a lot of the same dynamics as QE1, but that in general the peaks in the set are more damped, while the peaks of QE1 are often quite a bit higher than those of QT1. The prediction horizon of the FIR model is very similar to that of QE1. They often simultaneously start increasing or decreasing, when a peak or through of QT1 follows.

As the QE1 model estimates the parameters $\beta$ off-line and the OLS method estimates them recurrently, at each point in time, it is interesting to see how much the estimates of $\beta$ of the OLS method vary over time, as the results are somewhat similar. The $\beta$ of the OLS method are presented in Appendix B-2-1. It can be seen that the estimated parameters of all four of the input variables, vary quite a bit over time. The intercept $\hat{\beta}_0$, on the other hand, is virtually zero, throughout the whole dataset.

One reason for the varying value of $\hat{\beta}$ could be the rolling window size. If this was increased this could cause the values to become more constant. This is evaluated by performing the same OLS method on the same dataset with a prediction horizon of 20, but changing the size of the rolling window from 500 to 2000. The resulting values of $\hat{\beta}$ are presented in
Appendix B-2-2. It can be seen that the values of $\hat{\beta}$ become less volatile, i.e. their change over time is smaller, but that the absolute values still see a large change over the data set. E.g. $\hat{\beta}(2)$ stay between 1 and 2, for the large part of the dataset, but around December 17th, increases to over 4. At practically the same time $\hat{\beta}(1)$ drops, from an average of -2, to -5. On the whole it can be said that increasing the rolling window from 500 to 2000, makes the model more stable, but the resulting performance in terms of $\text{MSPE}_{norm}$ increases from 0.44 for the OLS method with a rolling window of 500, to 0.65 for the OLS method with a rolling window of 2000.
The goal of this research is to assess whether a procedure can be determined which constructs prediction models that are able to produce accurate predictions on some output variable of a process, without use any prior knowledge about the process. If no prior knowledge about the process is used, this means that the procedure has no knowledge about which of the available input variables contains useful information and which do not.

Therefore all possible input variables must be used by the regression method, and it must determine itself, which of the predictors that contain the useful information. This chapter describes a number of methods that are able to deal with this problem. Two different approaches are discussed; regularisation methods, and methods using derived inputs. In addition, some data pre-processing procedures are discussed, which are performed to the measured data.

5-1 Regularisation methods

Regularization methods are one method of dealing with a large number of input variables. Two different regularization methods are discussed, the ridge regression and the least absolute shrinkage and selection operator (LASSO).

5-1-1 Ridge regression

The ridge regression applies regularisation to the coefficients, by adding a penalty term to the least squared estimation. The ridge coefficients are determined with a penalized residual sum of squares;

\[
\hat{\beta}_{\text{ridge}} = \arg \min_{\beta} \left\{ \sum_{t=1+h}^{N} (y_t - \beta_0 - \sum_{j=1}^{p} x_{t-h,j} \beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\} \tag{5-1}
\]

Here \( \lambda \) is a complexity parameter that controls the amount of shrinkage. The intercept \( \beta_0 \) has been left out of the penalty term. It is estimated by taking the average of the outputs;
\[ \hat{\beta}_0 = \frac{1}{N-h} \sum_{t=1}^{N-h} y_t. \]

In matrix form the unique solution for the estimate of \( \beta \) is written as:

\[ \hat{\beta}_{\text{ridge}} = (X^T X + \lambda I)^{-1} X^T y \]  

(5-2)

The idea of penalizing by the sum-of-squares of the parameters is also used in neural networks, where it is known as weight decay. The variables have to be normalised before this method is applied.

### 5-1-2 LASSO method

The least absolute shrinkage and selection operator (LASSO) is a regularization method like the ridge regression, with subtle but important differences. It was first proposed in [7]. The lasso estimate is defined by:

\[
\hat{\beta}_{\text{lasso}} = \arg \min_{\beta} \left\{ \frac{1}{2} \sum_{t=1}^{N} (y_t - \beta_0 - \sum_{j=1}^{p} x_{t-h,j} \beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\} 
\]  

(5-3)

\( \beta_0 \) is estimated in the same manner as it was estimated for the ridge regression; by taking the average of \( y \) and thereafter fitting a model without an intercept.

The difference between the ridge regression and the lasso, is that in (5-3) the ridge penalty term \( \sum_{j=1}^{p} \beta_j^2 \) is replaced with the lasso penalty term \( \sum_{j=1}^{p} |\beta_j| \). The latter constraint makes the solutions non-linear in the \( y \), and there is no closed form expression as in the ridge regression. Therefore computing the lasso solution is a quadratic programming problem, and the necessary computation time for the LASSO is much longer than that of the ridge regression. Another result of the difference in penalty term is that with the LASSO the estimates of \( \beta_j \) can actually be set to zero, whereas with the ridge regression it can be set close to zero, but never actually zero.

The identification of the finite impulse response (FIR) models for both the ridge regression and the LASSO is done in an equivalent manner as by the ordinary least squares (OLS). The only differences are the addition of the penalty term in the calculation of the estimate of \( \beta \) and the calculation of the intercepts \( \beta_0 \).

### 5-2 Methods using derived inputs

Other methods that can be applied to situations, where a large number of input variables are, are methods using derived inputs. These are especially useful when the input variables are highly correlated. The general scheme can be described as followed; a method is used to produce a small number of linear combinations \( z_m, m = 1, \ldots, M \), of the original inputs \( x_j, j = 1, \ldots, p \). These linear combinations \( z_m \) are then used as inputs for the regression models, instead of \( x_j \). The methods differ in the manner in which the linear combinations are constructed. Two of these methods are discussed in this report, the principal component regression and the partial least squares.
5-2-1 Principal components regression

The principal components analysis (PCA) is a statistical procedure that can be used for many different applications. The general idea is that an orthogonal transformation is used to convert a dataset of correlated variables into a set of linearly uncorrelated variables. These uncorrelated variables are called the principal components (PCs).

The PCA can also be applied to the regression problem, this is known as the PC regression. This method uses the PCs of the predictor variables as inputs instead of the predictor variables themselves. If each of the PCs is included in the regression, the resulting output is the same as when it would be constructed with the original, untransformed dataset. However if a number of the PCs are excluded from the regression, estimators are obtained that are usually biased, but simultaneously greatly reduce the variance of the output [8].

The PC regression procedure is as follows: let us take matrix $X$, which consists of $p$ columns of predictors $x_j$ where $j = 1, \ldots, p$, and each column has $N$ rows, corresponding to the measurements at each time step. The algorithm for constructing predictions with PCA is as follows:

1. Take the input variables $x_j$, for $j = 1, \ldots, p$, which span $N$ measurements.
2. Deduct the averages from each column:
   $$x_j^c = x_j - \frac{1}{N} \sum_{t=1}^{N} x_{t,j}$$  \hfill (5-4)
3. The singular value decomposition (SVD) is applied to the centred input matrix $X^c$
   $$X^c = UDV^T$$  \hfill (5-5)
   where $U$ and $V$ are $N \times p$ and $p \times p$ orthogonal matrices, with the columns of $U$ spanning the column space of $X$, and the columns of $V$ spanning the row space. $D$ is a $p \times p$ diagonal matrix, with diagonal entries $d_1 \geq d_2 \geq \ldots \geq d_p \geq 0$ called the singular values of $X$. The SVD of the $X^c$ is another way of expressing the PCs of the variables in $X$.
   The columns of $V$ are the eigenvectors $v_j$ are also called the PC directions of $X$.
4. Determine the derived input variables;
   $$Z = XV$$  \hfill (5-6)
   where $X = (x_1, \ldots, x_p)$, $Z = (z_1, \ldots, z_p)$. The first PC direction $v_1$ has the property that $z_1 = X v_1$ has the largest sample variance amongst all normalised linear combinations of the columns of $X$. The derived variable $z_1$ is called the first PC of $X^c$.
   Subsequent PCs $z_j$ have maximum variance, subject to being orthogonal to the earlier ones.
5. Choose $M$ such that it fulfils $M \leq p$
6. Use the following model for a FIR model;
   $$y_t = \theta_0 + \sum_{m=1}^{M} \theta_m z_{t-h,m} + \varepsilon_t$$  \hfill (5-7)
   where $\theta_0$ is the intercept and is calculated as the average of the outputs, and $\theta_m$, $m = 1, \ldots, M$ are the the parameters.
5-2-2 Partial least squares regression

Like PCA, partial least squares (PLS) also construct a set of linear combinations of the inputs for regression. However, contrary to PCA, PLS also uses the output $y$ for this construction. The procedure is described by the following algorithm:

1. Determine $\hat{y}^{(0)} = y$ and $\hat{x}_j^{(0)} = x_j$, for $j = 1, \ldots, p$
2. For $m = 1, \ldots, p$
   (a) Determine the derived input $z_m$ as follows:
   \[
   z_m = \sum_{j=1}^{p} \hat{\phi}_{m,j} x_j^{(m-1)}
   \]  
   where $\hat{\phi}_{m,j} = \sum_{t=1}^{N} x_{t,j}^{(m-1)} y_t$
   (b) Orthogonalize each $x_j^{(m-1)}$ w.r.t. $z_m$:
   \[
   x_j^{(m)} = x_j^{(m-1)} - \left[ (z_m z_m)^{-1} z_m x_j^{(m-1)} \right] z_m
   \]
   for $j = 1, \ldots, p$
3. Choose $M$ such that it fulfils $M \leq p$
4. Use the following model for a FIR model;
   \[
   y_t = \theta_0 + \sum_{m=1}^{M} \theta_m z_{t-h,m} + \varepsilon_t
   \]  
   where $\theta_0$ is the intercept, which is also estimated by the PLS regression, and $\theta_m$, $m = 1, \ldots, M$ are the the parameters.

Both the methods PCA and PLS have been described in the setting where their derived input variables $z_m$ are used as inputs for a FIR, however it is also possible to use these variables as inputs for other prediction methods.

5-3 Data pre-processing

5-3-1 Data normalisation

Before any of the regression methods are applied to the data, the data is normalised. Every time a window of data is taken to produce an out-of-sample prediction, normalisation is applied to the data within the window. This is done as follows:

\[
\begin{align*}
x_{j,\text{norm}} &= \frac{x_j - \mu_j}{\sigma_j} \\
\end{align*}
\]

D.W.B. Bender

CONFIDENTIAL

Master of Science Thesis
where $\mu_j$ and $\sigma_j$ are the mean and standard deviation of $x_j$ in the window. Equivalently, the output $y$ is also normalised and this generates the set $y_{\text{norm}}$:

$$y_{\text{norm}} = \frac{y - \mu_y}{\sigma_y}$$ (5-12)

After $x_{j,\text{norm}}$, $j = 1, \ldots, p$, and $y_{\text{norm}}$ are used in the regression model, and the normalised out-of-sample estimate of the output at time $t$ is generated, $\hat{y}_t_{\text{norm}}$, a reverse normalisation is applied to transform the normalised out-of-sample output estimate into the regular out-of-sample output estimate;

$$\hat{y}_t = (\hat{y}_{t,\text{norm}} \times \sigma_y) + \mu_y$$ (5-13)

The reason that the normalisation is applied at each time step separately, as opposed to applying normalisation to the entire dataset straight away, is that this way only information is used that would have been available at each specific point in time. The predictions that are made using these methods, therefore remain completely out-of-sample.

### 5-3-2 Removing erroneous measurement data

In addition to normalisation, the data is also pre-processed by deleting erroneous predictor measurement sets from the data. Measurements sets can be erroneous for a number of reasons, one of which is a malfunctioning of the sensor, causing to not send new data to the data control system. In this case a zero-order hold is applied to the value by the control system. Predictor data that shows such behaviour, remaining constant for an extended period of time, is eliminated from the data.

There are numerous other reasons that measurement data can be erroneous, e.g. a bias or a drift enter into the sensor measurements. In this research the assumption is made the input and output measurements, that is not eliminated for constant behaviour, is an accurate enough approximation of the true values of the variables that the regression methods can be applied, i.e. no other data pre-processing is applied to the data.
Chapter 6

Method Analysis

Each of the regression methods that were described in the previous chapter have at least one parameter which must be defined before it is implemented. To assess what values should be used for these parameters, an analysis is performed for each prediction methods using the data for the period December 10th through December 19th 2015. All 36 available variables in the Depentanizer (C-X11) subsystem are used as inputs for the regression methods.

6-1 Regularisation methods

Both regularisation methods only require determine what their respective parameter $\lambda$ should be. To determine this, both methods are tested for varying values of $\lambda$ and their performances are compared.

6-1-1 Ridge regression

In the ridge regression the complexity parameter $\lambda$ must be determined. To assess the performance of different values of $\lambda$ the ridge regression is performed for the analysis period using $\lambda = 0, 0.01, \ldots, 1.99, 2$. The results for each of the regression methods is shown in Figure 6-1. A rolling window of 500 and a prediction horizon of 20 were used.

The MSPE$_{norm}$ decreases when the $\lambda$ increases from zero until around 0.7. It is therefore decided that the ridge regression should be implemented using a $\lambda$ of 0.7. This will regularise the use of many parameters somewhat, but will probably not completely restrict the use of many predictors, if they seem to contain useful information.
6-1-2 LASSO

The second regularization method that is used is the least absolute shrinkage and selection operator (LASSO). Like with the ridge regression only the parameter $\lambda$ has to be determined. The computation of the LASSO is, however, much longer than that of the ridge regression, as it is a non-linear optimisation problem.

The analysis is performed similarly to the analysis of the ridge regression, by applying the LASSO to the analysis dataset and using different values of $\lambda$, varying between $10^{-5}$ and $10^{-1}$. The results are presented in Figures 6-2a and 6-2b. For the analysis a rolling window of 500 and a prediction horizon of 20 were used.

Where the ridge regression only performs regularisation, LASSO performs both regularisation
and feature selection. This means that when LASSO estimate parameters, it also sets some of these parameters to zero, which means that the corresponding input variable is excluded from the model. As the LASSO constructs new estimates at each sample of dataset the amount of excluded variables can vary for each sample. In Figure 6-2b the range of included variables for the entire dataset is indicated for each different \( \lambda \).

It can be seen that the smallest MSPE \text{norm} occurs at a \( \lambda \) of 0.027. This method includes between 16 and 33 input variables throughout the entire dataset. In the further implementation of LASSO a \( \lambda \) of 0.027 will be used.

### 6-2 Methods using derived inputs

For the analysis of the two methods using derived inputs the number of included derived inputs need to be determined. The inputs of the entire sets are transformed, and the variances, of respectively the inputs themselves and the output, that can be explained are examined.

#### 6-2-1 Principal components regression

For the analysis for the PC regression a principal components analysis (PCA) is performed on all of the input variables for the entire analysis data set. The principal component (PC)s are ordered according to how much of the total variance they explain. Figure 6-3 presents the results of the analysis. The cumulative fraction of explained variance is shown for \( M \) PCs.

![Figure 6-3: Results of the PCA](image)

It is clear that using 15 PCs, one is able to describe practically all variance in the input variables. Using only ten PCs, around 95% of the variance explained, which is close to 100%. Using just 5 PCs, explains around 81% of the variance. Reducing the amount of PCs included from 15 to ten or even five, might make the models more robust. Therefore in the further implementation of PC regression, 5, 10, and 15 PCs are used.
6-2-2 Partial least squares regression

In contrast to PCA, the partial least squares (PLS) does not only look at the input variables, but also look at the output variable. Therefore, for the analysis of the PLS regression, both the input and output variables for the entire dataset are used. The PLS analysis looks how much of the variance in the output variable can be explained by the input variables. For this analysis a prediction horizon has to be set as we want to know how much of the future output variance can be explained by the current input variables. The prediction horizon is set to 20 for the analysis. The PLS analysis is performed, and the PLS components are ranked according to how much of the total variance of the output the explain. Figure 6-4 presents the results of this analysis. The cumulative fraction of explained variance of the output is shown for $M$ PLS components, where $M = 1, \ldots, 35$.

![Figure 6-4: Results of the PLS analysis](image)

The total variance of the output that can be explained by the input variables is around 90%. This is achieved after about 22 PLS components are used. 15 components are able to explain around 85% of the variance and ten are able to explain about 78%. The examine whether including less components makes the models more robust, in the implementation 10, 15, and 20 PLS components are used.
Chapter 7

Results

Once the parameters for each of the regression methods are determined, they are applied to the two identified datasets of the Depentanizer (C-X11), the normal operations dataset (NOD) and the abnormal operations dataset (AOD). In this chapter the results of this are evaluated and compared.

In total 10 regressions are compared; the first is ordinary least squares (OLS) for the finite impulse response (FIR) method using the 4VAR as input variables, which is used as a benchmark for the other methods. The remaining nine are: OLS, ridge regression with $\lambda = 0.7$, and least absolute shrinkage and selection operator (LASSO) with $\lambda = 0.0027$ are applied to all 34 input variables without any form of regularization or dimension reduction, principal component (PC) regression is applied with 5, 10 and 15 PCs, and finally partial least squares (PLS) is applied using 10, 15 and 20 partial least squares components (PLSCs). An overview is given in Table 7-1.

<table>
<thead>
<tr>
<th>Regression method</th>
<th>Predictors</th>
<th>Parameters</th>
<th>#</th>
<th>Abbreviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ordinary least squares</td>
<td>4</td>
<td>1</td>
<td>ols1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>36</td>
<td>2</td>
<td>ols</td>
<td></td>
</tr>
<tr>
<td>Ridge regression</td>
<td>36 $\lambda = 0.7$</td>
<td>3</td>
<td>ridge0.7</td>
<td></td>
</tr>
<tr>
<td>LASSO</td>
<td>36 $\lambda = 0.0027$</td>
<td>4</td>
<td>lasso0.0027</td>
<td></td>
</tr>
<tr>
<td>Principal component regression</td>
<td>36 5 PCs</td>
<td>5</td>
<td>pca5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>36 10 PCs</td>
<td>6</td>
<td>pca10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>36 15 PCs</td>
<td>7</td>
<td>pca15</td>
<td></td>
</tr>
<tr>
<td>Partial least squares</td>
<td>36 10 PLSCs</td>
<td>6</td>
<td>pls10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>36 15 PLSCs</td>
<td>9</td>
<td>pls15</td>
<td></td>
</tr>
<tr>
<td></td>
<td>36 20 PLSCs</td>
<td>10</td>
<td>pls20</td>
<td></td>
</tr>
</tbody>
</table>
7-1 Normal operations dataset

The discussed regression methods are implemented to the NOD, which was discussed in Chapter 3 and represents normal operations of the C-X11. The dataset of QTXX1 (QT1) for this period is shown in Figure 7-1.

![Figure 7-1: Normal operations dataset of QT1](image)

Each regression method is applied with three different rolling window sizes, namely 300, 500, and 1000 samples. In addition, each method is evaluated for thirty different prediction horizons; 1 through 30 minutes. This means that each of the ten regression methods is applied 90 times. The results of all these implementations are given in Figure 7-2.

It can be observed that for each of the three rolling windows the three PLS methods have a practically identical performance in terms of MSPE$_{norm}$. It can also be seen that their MSPEs$_{norm}$ for each prediction horizon are equal or greater than one, which means that they perform either the same or worse than the historical mean. It can thus be concluded that the PLS methods are not able to predict the behaviour of the process from the input variables.

The PC regression methods that makes use of five PCs performs slightly better that the benchmark OLS method for prediction horizons smaller than about ten minutes, for each prediction horizon. However when the prediction horizon becomes larger it performs worse than the benchmark. The PC regression which used 10 PCs, outperforms the original OLS method with four input variable, for each of the rolling windows, and a prediction horizon of lower than 15. Once the prediction horizon becomes 15 or large, the PC regression performs very similar, and in the case where the rolling window is 300, it even performs slightly worse. However the PC regressions which use 15 PCs, perform better than both the other PC regressions and the original OLS. In each case, the MSPEs$_{norm}$ are a lot lower for shorter prediction horizons and they increase and converge to each other slowly, as the prediction horizon increases. Each of the PC regressions perform better at lower prediction horizons with the smaller rolling windows, 300 and 500, than with a rolling window of 1000. Oppositely when the prediction horizon is larger, a rolling windows of 300, produces worse predictions. For the steady state situation, if enough PCs are used, PC regressions can be used to make predictions.
Figure 7-2: Results of the regression methods for the NOD
that are more accurate than those generated by the OLS with four input variables. Thus, it can be concluded that PC regressions can be used to make accurate prediction methods of QT1 using the entire set of input variables and no prior knowledge about the process.

Finally there are three methods which have a extremely similar performance, and outperform all the others in terms of MSPE\textsubscript{norm}. They are able to continue producing accurate forecasts for longer prediction horizons. These are the OLS, using all 34 input variables, the ridge regression, and the LASSO. When a rolling window of 1000 is used, each of these methods produces predictions, which outperform all the other methods for each prediction horizon and rolling window. The optimal performance is achieved with a rolling window of 1000 and a prediction horizon of 14, where the OLS has a MSPE\textsubscript{norm} of 0.091.

In Table 7-2, the results of each of the methods for a prediction horizon of 20 are presented. The computation time of the prediction set is indicated by $t_c$. It is clear the OLS, the PC regression, the ridge regression, and the LASSO methods, each have a relatively low MSPE\textsubscript{norm}. The computation time of the methods is also shown. It can be seen that the PC regression and PLS methods have a relatively long computation time compared to the OLS and ridge regression methods. The computation time of the PC regression, remains practically the same when the number of PC methods is increased. However, the computation time of the PLS regression methods increases significantly when the amount of PLS components is increased. For both the PC regression and the PLS regression methods it can be seen that the computation time increases when the rolling window is increased. In the case of the OLS and the ridge regression method this is also occurs, however this increase is practically negligible.

As mentioned in Chapter 5, the LASSO method has to be solved as a quadratic optimisation problem. The results of this can be clearly seen in the computation time of the methods.
generated by the LASSO regression method. The computation time, when a rolling window of 300 is taken, of the LASSO method is 299 seconds, while that of the equivalent ridge regression method is 5 seconds. When the rolling window increases, the computation time of the LASSO methods increases even more, to 426 seconds for a rolling window of 500, and even 957 seconds for the largest rolling window. As the results of the LASSO are practically identical to the results of the ridge regression, it can be concluded, that the ridge regression performs more favourably than the LASSO, as the computation time is significantly smaller.

To illustrate the performance of the prediction methods, Figure 7-3a shows the prediction set of the linearly interpolated output set QT1, and the predicted datasets generated with OLS, ridge regression, and LASSO methods. The predicted sets are constructed with a rolling window of 1000 samples and a prediction horizon of 20 minutes. Figure 7-3b shows the same datasets for subset of the NOD during the 16th of December 2015. Finally Figure 7-3b shows only the ridge regression prediction set, and includes bounds, which are the blue lines and indicate the predicted value ±σt. At each sample, σt is determined as the variance of the prediction errors of the previous 300 predictions.

It can be seen in Figure 7-3b, that the three prediction methods, show practically identical dynamics. The only difference is in the height of the peaks. In general it can be seen that the LASSO method has the lowest peaks, than the ridge regression method, and finally the OLS method has the highest peaks. It can be seen that the prediction methods are able to almost completely predict the the behaviour of the real output 20 minutes before it occurs. By examining Figure 7-3c, it can be seen that the real value of QT1, is for a large part of the dataset, within one standard deviation of the predicted value.

It is interesting that the results are similar, as the only differences between the OLS and the ridge regression and LASSO, is that latter two add a penalty term when constructing estimates of β. This could mean that the regularization performed by the penalty terms in both methods is very small. However if the β’s of the OLS and ridge regression methods are examined throughout the dataset, it can be observed that this is not the case. These β’s are presented in Appendix C-1-1. A number of the β’s of the ridge regression are clearly smaller than those of the OLS, which means that the regularisation definitely affects the regression methods. Figure 7-4 shows the amount of included variables of each method that is constructed by the LASSO regression method with a rolling window of 1000 and a prediction horizon of 20. It can be seen that throughout the dataset, each method includes between 17 and 31 predictors, which is always at least half of the available predictors. This is consistent with the fact the performance of the ridge regression and the LASSO methods are very similar to that of the OLS method, as the ridge regression and LASSO are both just a regularised version of the OLS. If LASSO would exclude more predictors it would be illogical that the performance of the OLS without regularisation, would perform so similarly.

It is possible that in the case of a steady state dataset the differences between the OLS, ridge regression, and LASSO methods are small, while for a dataset where a fault occurs, the response of the methods could be very different. This will be investigated, when the regression methods are applied to AOD.
Figure 7-3: Linearly interpolated dataset of QT1 and the predicted output sets for the NOD. The green dotted lines indicate the equivalent periods in time.
In Section 4-4-1 the use of concatenated variables as inputs for the regression methods was discussed. This was implemented in Section 4-5 for the analysis of the different interpolation methods. The results were very close together, and it could be concluded that in that implementation using concatenated variables did not improve the performance. However, that implementation only included four of the input variables of the C-X11 and it was only done using OLS and none of the other regression methods. Therefore an extra analysis is done to see whether using concatenated variables can improve the performances of the best performing regression methods. The regression methods are applied to ten sets of input variables, corresponding to 0 up to 9 concatenated variables. Each regression is performed using the OLS and the ridge regression method, using rolling window size of 1000, and prediction horizons of 1 through 30 minutes. The results of the ridge regression methods are presented in Figure 7-5 and those of the OLS can be found in Appendix C-1-2. The amount of concatenated variables for each regression method is indicated by its subscript.

It can be observed straight away that including concatenated variables can improve the performance of the regression methods, in terms of MSPE$_{norm}$. For a prediction horizon of 20 minutes, the MSPE$_{norm}$ is decreased by half. If the prediction horizon is increased further, the differences become even larger. The results of the regression methods for a prediction horizon of 20 minutes are presented in Table 7-3.

By using five concatenated variables, the MSPE$_{norm}$ of the ridge regression is decreased by half in comparison to using no concatenated variables. The same goes for the OLS methods and the results if the best performing OLS and ridge regression methods are practically the same. There seems to be an optimum in terms of MSPE$_{norm}$ around seven concatenated variables, for both regression methods. Increasing the amount of concatenated variables beyond this start to increase the MSPE$_{norm}$ again. The computation time does increase by a factor when more concatenated variables are included in the regression method. However, a computation time of less than 200 seconds for the entire dataset is still easily acceptable. It can be concluded that adding concatenated variables to the inputs can significantly improve
Figure 7-5: The results of the regression methods using concatenated variables for the NOD

Table 7-3: Results of the regression methods using concatenated variables for NOD

<table>
<thead>
<tr>
<th>method</th>
<th>MSPE\textsubscript{norm}</th>
<th>$t_c$</th>
<th>method</th>
<th>MSPE\textsubscript{norm}</th>
<th>$t_c$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ols0</td>
<td>0.128</td>
<td>9</td>
<td>ridge0.70</td>
<td>0.128</td>
<td>9</td>
</tr>
<tr>
<td>ols1</td>
<td>0.088</td>
<td>18</td>
<td>ridge0.71</td>
<td>0.091</td>
<td>15</td>
</tr>
<tr>
<td>ols2</td>
<td>0.078</td>
<td>31</td>
<td>ridge0.72</td>
<td>0.078</td>
<td>25</td>
</tr>
<tr>
<td>ols3</td>
<td>0.069</td>
<td>58</td>
<td>ridge0.73</td>
<td>0.069</td>
<td>52</td>
</tr>
<tr>
<td>ols4</td>
<td>0.063</td>
<td>76</td>
<td>ridge0.74</td>
<td>0.062</td>
<td>69</td>
</tr>
<tr>
<td>ols5</td>
<td>0.058</td>
<td>97</td>
<td>ridge0.75</td>
<td>0.058</td>
<td>90</td>
</tr>
<tr>
<td>ols6</td>
<td>0.057</td>
<td>116</td>
<td>ridge0.76</td>
<td>0.055</td>
<td>111</td>
</tr>
<tr>
<td>ols7</td>
<td>0.056</td>
<td>134</td>
<td>ridge0.77</td>
<td>0.054</td>
<td>127</td>
</tr>
<tr>
<td>ols8</td>
<td>0.057</td>
<td>156</td>
<td>ridge0.78</td>
<td>0.054</td>
<td>153</td>
</tr>
<tr>
<td>ols9</td>
<td>0.061</td>
<td>186</td>
<td>ridge0.79</td>
<td>0.056</td>
<td>179</td>
</tr>
</tbody>
</table>
the performance of the regression methods, at least for normal operating conditions.

7-2 Abnormal operations dataset

The same ten regression methods are applied to a second dataset. Even though some of the methods did not perform well, and some performed exceptionally well, they are all included, to see whether this same results is gained when a dataset is used, which does not remain at steady state. The dataset is the abnormal operations dataset (AOD), which represent abnormal operating conditions. The dataset of QT1 for this period is shown in Figure 7-6.

![Figure 7-6: Abnormal operations dataset of QT1](image)

It can be seen that for the first eleven days the process remains at normal operations, with two minor disturbances on the fourth and fifth day. However, at the end of the 11th of January it can be seen that the process leaves this normal operating state. The mass fraction drops from roughly 3.5 to almost 1. After half a day, the process recovers, but only a few hours later QT1 again drops, this time further even than 1. This dataset is used to approximate a dataset where a sudden fault enters the system and to determine whether, by recurrently identifying the FIR model online, the regression methods are able to predict such behaviour. The idea being, that when a fault starts to appear, the method will learn it and will be able to predict its later, probably more detrimental, effect.

As with the previous dataset, each regression is applied with the three rolling window sizes of 300, 500, and 1000 samples and thirty prediction horizons, 1 through 30 minutes. The results are presented in Figure 7-7. As with the previous dataset, the results of the PLS regressions are practically identical to each other, and significantly worse than the other methods. Therefore their results have been excluded from Figure 7-7.

The PC regression methods show a fairly similar behaviour as with the NOD. For practically all rolling windows and prediction horizons, the PC regression using five PCs is outperformed by the PC regression using 10 PCs, and likewise, it, in turn is outperformed by the PC regression using 15 PCs. In general, increasing the rolling window does slightly decreases
Figure 7-7: Results of the regression methods for the AOD.
the MSPE\textsubscript{norm}, for the PC regression methods with 10 and 15 PCs, and slightly increases the MSPE\textsubscript{norm} of the PC regression using five PCs. The performances of the OLS, using all predictors, the ridge regression, and the LASSO methods are again incredibly similar. They outperform all the other regression methods in terms of MSPE\textsubscript{norm}. The only exception is the OLS method using four inputs, for the smaller two rolling windows and higher prediction horizons.

The results of all the regression methods with a prediction horizon of 20 minutes are presented in Table 7-4. The OLS method using all predictors, the ridge regression method, and the LASSO methods, each outperform or perform equally, when the rolling window size is 500, as the OLS method using the four predictors. The results of the PC regression methods with 15 PCs, also performs nicely, but in each case performs slightly worse than the regression methods just discussed.

Examining the computation times, it can be concluded that the only difference with the previous dataset, is that the computation times are all longer. This is logical as this dataset is larger than the previous dataset. The computation times of the different regression methods remain the same in relative comparison to each other, which was expected, as a different dataset with the same amount of inputs and outputs, should not change the computation time of the regression methods. The performances of these methods are also, again practically identical to the results of the OLS and ridge regression methods. Therefore it can be concluded that the LASSO does not have any advantages over the ridge regression in this implementation. It is still possible that for some other dataset, the regularisation part of LASSO could cause it to perform better than the OLS method, but it is unlikely to perform much better than the ridge regression method, as this has a similar regularisation factor.

Table 7-4: Results of the regression methods for the AOD

<table>
<thead>
<tr>
<th>method</th>
<th>rw</th>
<th>MSPE\textsubscript{norm}</th>
<th>tc</th>
<th>method</th>
<th>rw</th>
<th>MSPE\textsubscript{norm}</th>
<th>tc</th>
</tr>
</thead>
<tbody>
<tr>
<td>ols4</td>
<td>300</td>
<td>0.34</td>
<td>4</td>
<td>ols4</td>
<td>500</td>
<td>0.34</td>
<td>4</td>
</tr>
<tr>
<td>ols</td>
<td>300</td>
<td>0.30</td>
<td>10</td>
<td>ols</td>
<td>500</td>
<td>0.23</td>
<td>12</td>
</tr>
<tr>
<td>ridge0.7</td>
<td>300</td>
<td>0.29</td>
<td>10</td>
<td>ridge0.7</td>
<td>500</td>
<td>0.22</td>
<td>12</td>
</tr>
<tr>
<td>lasso0.0027</td>
<td>300</td>
<td>0.29</td>
<td>665</td>
<td>lasso0.0027</td>
<td>500</td>
<td>0.22</td>
<td>919</td>
</tr>
<tr>
<td>pca5</td>
<td>300</td>
<td>0.61</td>
<td>44</td>
<td>pca5</td>
<td>500</td>
<td>0.61</td>
<td>50</td>
</tr>
<tr>
<td>pca10</td>
<td>300</td>
<td>0.41</td>
<td>44</td>
<td>pca10</td>
<td>500</td>
<td>0.44</td>
<td>50</td>
</tr>
<tr>
<td>pca15</td>
<td>300</td>
<td>0.36</td>
<td>44</td>
<td>pca15</td>
<td>500</td>
<td>0.31</td>
<td>51</td>
</tr>
<tr>
<td>pls10</td>
<td>300</td>
<td>1.14</td>
<td>31</td>
<td>pls10</td>
<td>500</td>
<td>1.19</td>
<td>35</td>
</tr>
<tr>
<td>pls15</td>
<td>300</td>
<td>1.15</td>
<td>47</td>
<td>pls15</td>
<td>500</td>
<td>1.20</td>
<td>54</td>
</tr>
<tr>
<td>pls20</td>
<td>300</td>
<td>1.19</td>
<td>61</td>
<td>pls20</td>
<td>500</td>
<td>1.21</td>
<td>70</td>
</tr>
<tr>
<td>ols4</td>
<td>1000</td>
<td>0.38</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ols</td>
<td>1000</td>
<td>0.18</td>
<td>18</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ridge0.7</td>
<td>1000</td>
<td>0.17</td>
<td>17</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lasso0.0027</td>
<td>1000</td>
<td>0.18</td>
<td>1722</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pca5</td>
<td>1000</td>
<td>0.73</td>
<td>64</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pca10</td>
<td>1000</td>
<td>0.43</td>
<td>65</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pca15</td>
<td>1000</td>
<td>0.32</td>
<td>65</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pls10</td>
<td>1000</td>
<td>1.59</td>
<td>44</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pls15</td>
<td>1000</td>
<td>1.58</td>
<td>64</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pls20</td>
<td>1000</td>
<td>1.59</td>
<td>85</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The performance of the best performing regression methods with a rolling window of 1000
and a prediction horizon of 20 are presented in Figure 7-8a. Figure 7-8b presents the subset of the AOD during the 11th and 12th of January 2016. The presented regression methods are the OLS, the ridge regression, and the LASSO. In Figure 7-8c the same ridge regression predicted set is present, with bounds depicting the predicted values $\pm \sigma_t$.

It can be observed that the predicted datasets are very similar. Each is able to predict the behaviour of the output quite accurately. When the sudden decrease comes the methods are able to predict this, and when the behaviour remains different from the steady state, the methods are able to still capture most of the dynamics. As with the NOD, the real output set is within $\sigma_t$ of the ridge regression predictions, for most of the subset of time. Around 21:00 on January 11th the fault enters the system and the values of QT1 begin to decrease. The predictions of the ridge regression do not predict this instantly and the real value passed through the standard deviation bound for a short period of time. But it can be seen that the ridge regression quickly learns the fault and becomes able to predict its effect on the behaviour of the system.

In Appendix C-2 the amount of included predictors by the LASSO is evaluated, as well as the estimated parameters $\hat{\beta}$ for both the OLS and the ridge regression method throughout the dataset. Each is evaluated for a rolling window of 1000 and a prediction horizon of 20. The results are similar to that of the NOD. The amount of predictors included varies between 16 and 34, compared to 17 and 31 for the NOD. Comparing the estimated parameters, it can be seen that a number of the estimated parameters of the ridge regression differ from those of the OLS method, so there is a clear result of the regularisation factor in the ridge regression, without it reducing its performance.
Figure 7-8: The linearly interpolated output set of QT1 and the predicted datasets for the AOD. The green dotted lines indicate the equivalent periods in time.
7-2-1 Concatenated variables

To evaluate whether using concatenated variables as inputs for the regression has the same positive effects in the AOD as it had on the NOD, the OLS and ridge regression method are also applied with concatenated variables to the AOD. The regressions are again applied with a rolling window of 1000 samples and prediction horizons of 1 to 30 minutes.

The results of the ridge regressions are shown in Figure 7-9 and the results of the OLS are shown in Appendix C-2-3. It can be clearly seen that the use of concatenated variables has a similar, albeit smaller, effect for the AOD, as it had for the NOD. The MSPE\text{norm} decreases when concatenated variables are added, but the difference is relatively small. The same behaviour can be observed for the OLS methods.

![Figure 7-9: Results of the regression methods using concatenated variables for the AOD](image)

Table 7-5 gives an overview of all the methods for a prediction horizon of 20 minutes. For both the OLS and ridge regression methods there is a drop in MSPE\text{norm} after adding a single set of concatenated variables. The optimal number of concatenated variables is two for the AOD, which is different from the NOD, where it was seven. It may be that adding a number of concatenated variables, such as four, can make the regression methods more robust to sudden changes and possible outliers in the input and output data. To evaluate this, Figure 7-10 shows the prediction for the ridge regression methods with zero and four concatenated variables. It can be seen that the method with four concatenated variables, is less volatile, but it adapts to changes, such as the decrease caused by the fault, at least as fast as the method with zero concatenated variables. It is therefore concluded that adding concatenated variables is a good way to make the methods more robust.
Table 7-5: Results of the regression methods using concatenated variables for AOD

<table>
<thead>
<tr>
<th>Method</th>
<th>MSPE_{norm}</th>
<th>t_c</th>
<th>Method</th>
<th>MSPE_{norm}</th>
<th>t_c</th>
</tr>
</thead>
<tbody>
<tr>
<td>ols0</td>
<td>0.178</td>
<td>18</td>
<td>ridge0.7</td>
<td>0.171</td>
<td>17</td>
</tr>
<tr>
<td>ols1</td>
<td>0.162</td>
<td>37</td>
<td>ridge0.7</td>
<td>0.150</td>
<td>30</td>
</tr>
<tr>
<td>ols2</td>
<td>0.160</td>
<td>59</td>
<td>ridge0.7</td>
<td>0.146</td>
<td>48</td>
</tr>
<tr>
<td>ols3</td>
<td>0.163</td>
<td>110</td>
<td>ridge0.7</td>
<td>0.148</td>
<td>100</td>
</tr>
<tr>
<td>ols4</td>
<td>0.164</td>
<td>145</td>
<td>ridge0.7</td>
<td>0.149</td>
<td>136</td>
</tr>
<tr>
<td>ols5</td>
<td>0.165</td>
<td>190</td>
<td>ridge0.7</td>
<td>0.149</td>
<td>176</td>
</tr>
<tr>
<td>ols6</td>
<td>0.168</td>
<td>229</td>
<td>ridge0.7</td>
<td>0.152</td>
<td>216</td>
</tr>
<tr>
<td>ols7</td>
<td>0.172</td>
<td>264</td>
<td>ridge0.7</td>
<td>0.154</td>
<td>245</td>
</tr>
<tr>
<td>ols8</td>
<td>0.174</td>
<td>301</td>
<td>ridge0.7</td>
<td>0.156</td>
<td>301</td>
</tr>
<tr>
<td>ols9</td>
<td>0.177</td>
<td>356</td>
<td>ridge0.7</td>
<td>0.158</td>
<td>349</td>
</tr>
</tbody>
</table>

Figure 7-10: The linearly interpolated output set of QT1 and the predicted datasets
In the previous chapter, it has been shown that a number of regression methods were able to construct accurate predictions of the key variable QTXX1 (QT1) in the Depentanizer (C-X11) unit. They were able to do so without using any prior knowledge about the process. However each method was first analysed, different parameter settings were evaluated, and the optimal setting was chosen. Ideally we want the regression methods to be so robust, that they can be applied to a process without having to taking any of these prior steps first.

To verify whether the identified regression methods are robust and general procedures, or whether they have been fitted to the data in the C-X11, they will now be applied to data from a completely different subsystem, the Deisopentanizer (C-X12).

8-1 Deisopentanizer

The C-X12 is a distillation column, just like the C-X11. However, where the C-X11 splits pentane (C5) and heavier compounds, the C-X12 actually splits the C5 into isopentane (iC5) and n-pentane (nC5). The boiling points of iC5 and nC5 are much closer together than those of C5 and the heavier compounds. One result of these close boiling points, is that the C-X12 is classified as a superfractionator. A second result of this fact, is that the process is much slower, as it is more difficult to separate the compounds. The settling time of the C-X12 is in the order of half a day, whereas that of the C-X11 is around 75 minutes. In addition the C-X12 has four quality measurements instead of one, two in the top flow and two in the bottom flow. These quality measurements are described in Table 8-1.

In Appendix D-1 the datasets of each of the four output variables is presented for the period November 2015 until May 2016. To evaluate the performance of the regression methods on these outputs, period of time is identified, which represent fairly standard operations. Not all output variables have to remain at steady state for the entire time. The period spanning March 14th until March 21st of 2016 is chosen. Data from this period of time is henceforth referred to as standard operations dataset (SOD). In Figure 8-1 the data of each of the four quality measurement is presented for this period. It can be observed that each of the output
variables have fairly stable outputs for the time period, but there is enough variation to be able to examine the applicability of the regression methods.

The C-X12 contain 25 input variable sensors, which are similar, but not exactly the same as the input variable sensors in the C-X11. In Appendix A-1 an overview and description is given of the entire set of input variables that are available in the C-X12.

8-1-1 Prediction results

Initially we want to see whether the regression methods, that have been identified as performing well in the last chapter, can be applied to this new subsystem and its data, and produce accurate predictions about its outputs, without using any prior knowledge. Therefore, the ridge regression method is chosen to be applied to each of the four output variables separately.

For the regression methods applied to each of the four output variables, the same set of 25 input variables is used. The regression methods are applied with a rolling window of 1000 samples and a prediction horizon of 20 minutes, as these settings have proved to work well in the C-X11. A side note is that the C-X12 process is slower than the C-X11, so these regression methods might perform better with larger rolling windows and prediction horizons, but this will be investigated later in the chapter. First we are interested in seeing how well the regression methods perform, when exactly the same parameters are chosen, as where determined to be optimal for the C-X11. For this initial validation, concatenated variables are not included. They will however be included later in this chapter.

The four resulting forecast sets constructed using the ridge regression are shown in Figure 8-2. Looking at the predicted set of QTYY1 it can be seen that the regression model is able to predict some of the behaviour of the output set. However some dynamics take, take the regression method some time to realise and as a results it lags behind the real output set. In addition the predicted set seems fairy volatile, especially around peaks and troughs. Lastly the predicted set seems to overshoot quite a bit a number of times during the 17th and 18th of March, this is undesirable, especially if limit checking was to be used for fault detection.

The second output set for which a predicted set is generated is the QTYY2. Again, quite a bit of the behaviour is correctly predicted by the regression method, but as with the QTYY1, around the 17th and 18th of March there are a number of peaks and troughs, that do not occur in the real output set. It is possible that there is some anomaly in the input data, that the regression method is not able to deal with, that causes these sudden spikes. In a number of cases, the predicted model is not able to correctly anticipate a certain increase or decrease of the output value, and the predicted set lags behind the real output.
Figure 8-1: Data of the output variables of the C-X12 for the SOD
Figure 8-2: The linearly interpolated output sets of the four output variables of the C-X12 and the predicted datasets of each for the SOD
Both these discussed output sets represent a mass fraction in the bottom flow of the C-X12. The third output set is the QTYY3, which is the mass fraction of nC4 in the top flow of the C-X12. If the predicted output set is compared to the real output set, it can be seen that the regression method is able to almost exactly predict the behaviour of the output variable. With the exception of a few peaks around the 17th and 18th of March, where the predicted peak is lower than the actual peak, the predicted set is almost the exact same as the real output set.

The last output variable is the QTYY4, and it represents the mass fraction of nC5 in the bottom flow of the C-X12. The regression method is able to nicely predict most of the behaviour of the output variable, especially the sudden increases and decreases are nicely predicted. Again around the 17th of March, there are some disturbances in the predicted output set, that do not appear in the real output set.

8-1-2 Regression method analysis

As mentioned the regression methods that were applied to the outputs of the C-X12, were the methods that were found to be optimal for the C-X11. However as the dynamics and the type and amount of input and output variables are different in the C-X12, this does not necessarily mean that the same parameter settings are optimal for the C-X12 as were optimal for the C-X11.

As the goal of this research is to find a robust method which does not require prior knowledge of the system or a lot of tweaking of the parameters before it works, we will not repeat the entire model analysis for the C-X12, as was done for the C-X11. Therefore the parameters of the regression methods will not be changed. However, it is decided to test the regression methods for a number of different rolling windows and prediction horizons, as it is known that the settling time of the C-X12 is much longer than that of the C-X11.

In addition to the ridge regression method, the ordinary least squares (OLS) method is applied. To verify whether principal components analysis (PCA) is more robust than the OLS or ridge regression methods, three PC regression methods are also applied, with respectively, 5, 10, and 15 principal component (PC)s.

The results of these analyses can be found in Appendix D-2 and are discussed individually for each output in the following sections.

QTYY1

Examining the results of the regression methods for the QTYY1, it can be seen that, once again, the ridge regression and finite impulse response (FIR) method have very similar results and the best performance. In addition it can be seen that the performances of the methods using a rolling window of 1000 and 2000, are inferior for each prediction horizon to those of with a rolling window of 300 and 500. In fact if we look only at the results of the predicted sets constructed with ridge regression and a prediction horizon of 20 minutes, the MSPE$_{norm}$ is decreases from 0.44 to 0.16, when the rolling window is decreased from 1000 to 300. It can be further concluded that increasing the prediction horizon also increases the MSPE$_{norm}$ of the predicted sets of the QTYY1, and thus it does not seem that there is some optimal value
for the prediction horizon. The best performing ridge regression predicted set for a prediction horizon of 20 minutes is the one, which is constructed with a rolling window of 300 and it is shown in Figure 8-3. This regression method with these parameter settings is able to predict the behaviour of the output very nicely. Each sudden increase and decrease is captured, and with the exception of some overshoot in a few of the peaks and troughs, most of the behaviour is captured.

Figure 8-3: The linearly interpolated output set of QTYY1 and the best performing ridge regression prediction method for the SOD

QTYY2

The results of the predicted set of the QTYY2 are very similar to that of the QTYY1, as the performance is much better for the when the predictions were constructed with a rolling window of 300 or 500, than when they were constructed with a rolling window of 1000 or 2000. In each case the results of the FIR and the ridge regression are very similar, but in contrast to the other implementations until now, the PC regression methods using 15 PCs actually performs very similar as well. The performance again seems to deteriorate when the prediction horizon is increased, without there being some optimal horizon. Therefore we again examine all the results for a prediction horizon of 20. Like with the QTYY1, the best performance is found when a rolling window of 300 is taken, and the resulting forecast set is shown in Figure 8-4. The resulting predicted dataset has some overshoot on a number of peaks and troughs, and at the end of the 19th of March, there is a sudden drop, which does not occur in the real output set. Aside from this, the regression method is able to predict most of the dynamics in the output set.
Figure 8-4: The linearly interpolated output set of QTYY2 and the best performing ridge regression prediction method for the SOD

QTYY3

The predicted set of the QTYY3 for a rolling window of 1000 and a prediction horizon of 20, that was discussed in the beginning of this chapter was already very close to the real output set. Examining the results of the other regression methods is can be seen that the results of the PC regression perform poorly for practically all parameter settings. The results of the FIR and the ridge regression are again very similar. A rolling window of 1000 seems to have the best results. A prediction horizon of 20 seems optimal, as both the MSPE$_{\text{norm}}$ start increasing when the prediction horizon is increased further. However the MSPE$_{\text{norm}}$ are so small at a prediction horizon of 20, that increasing a bit still results in a very nice performance. This means that a prediction horizon of 30, and even 40, are still able to capture the dynamics fairly well. Both of these resulting prediction set are shown in is shown in Figure 8-5. It can be seen that the resulting predicted set are extremely similar. However when the prediction horizon is increased the overshoot also tends to be slightly larger, but in general the regression methods predict the behaviour accurately.
Figure 8-5: The linearly interpolated output set of QTYY3 and the best performing ridge regression prediction method for the SOD

QTYY4

Finally, the results of the predicted set of QTYY4 are evaluated. The FIR and the ridge regression perform better than the other methods overall. Decreasing the rolling window to 300 increases the performance of the prediction set, if the prediction horizon is 20. The result of this is shown in Figure 8-6. The behaviour of the output is predicted nicely.

Figure 8-6: The linearly interpolated output set of QTYY4 and the best performing ridge regression prediction method for the SOD

8-1-3 Concatenated variables

To evaluate whether using concatenated variables as inputs for the regression can improve the performance on the C-X12 output variables, the best performing ridge regression method
for each output variable is implemented again with one through nine concatenated variables. The results are presented in Table 8-2.

<table>
<thead>
<tr>
<th>output</th>
<th>model</th>
<th>rw</th>
<th>MSPE$_{norm}$</th>
<th>$t_c$</th>
<th>output</th>
<th>model</th>
<th>rw</th>
<th>MSPE$_{norm}$</th>
<th>$t_c$</th>
</tr>
</thead>
<tbody>
<tr>
<td>QTYY1</td>
<td>ridge0.7$_0$</td>
<td>300</td>
<td>0.144</td>
<td>3</td>
<td>QTYY2</td>
<td>ridge0.7$_0$</td>
<td>300</td>
<td>0.147</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>ridge0.7$_1$</td>
<td>300</td>
<td>0.124</td>
<td>5</td>
<td></td>
<td>ridge0.7$_1$</td>
<td>300</td>
<td>0.145</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>ridge0.7$_2$</td>
<td>300</td>
<td>0.116</td>
<td>8</td>
<td></td>
<td>ridge0.7$_2$</td>
<td>300</td>
<td>0.140</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>ridge0.7$_3$</td>
<td>300</td>
<td>0.111</td>
<td>10</td>
<td></td>
<td>ridge0.7$_3$</td>
<td>300</td>
<td>0.139</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>ridge0.7$_4$</td>
<td>300</td>
<td>0.109</td>
<td>15</td>
<td></td>
<td>ridge0.7$_4$</td>
<td>300</td>
<td>0.136</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>ridge0.7$_5$</td>
<td>300</td>
<td>0.106</td>
<td>18</td>
<td></td>
<td>ridge0.7$_5$</td>
<td>300</td>
<td>0.133</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>ridge0.7$_6$</td>
<td>300</td>
<td>0.104</td>
<td>22</td>
<td></td>
<td>ridge0.7$_6$</td>
<td>300</td>
<td>0.130</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>ridge0.7$_7$</td>
<td>300</td>
<td>0.102</td>
<td>26</td>
<td></td>
<td>ridge0.7$_7$</td>
<td>300</td>
<td>0.131</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td>ridge0.7$_8$</td>
<td>300</td>
<td>0.103</td>
<td>34</td>
<td></td>
<td>ridge0.7$_8$</td>
<td>300</td>
<td>0.135</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td>ridge0.7$_9$</td>
<td>300</td>
<td>0.103</td>
<td>38</td>
<td></td>
<td>ridge0.7$_9$</td>
<td>300</td>
<td>0.138</td>
<td>38</td>
</tr>
<tr>
<td>QTYY3</td>
<td>ridge0.7$_0$</td>
<td>1000</td>
<td>0.077</td>
<td>5</td>
<td>QTYY4</td>
<td>ridge0.7$_0$</td>
<td>300</td>
<td>0.112</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>ridge0.7$_1$</td>
<td>1000</td>
<td>0.066</td>
<td>7</td>
<td></td>
<td>ridge0.7$_1$</td>
<td>300</td>
<td>0.103</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>ridge0.7$_2$</td>
<td>1000</td>
<td>0.059</td>
<td>12</td>
<td></td>
<td>ridge0.7$_2$</td>
<td>300</td>
<td>0.096</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>ridge0.7$_3$</td>
<td>1000</td>
<td>0.055</td>
<td>18</td>
<td></td>
<td>ridge0.7$_3$</td>
<td>300</td>
<td>0.093</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>ridge0.7$_4$</td>
<td>1000</td>
<td>0.052</td>
<td>24</td>
<td></td>
<td>ridge0.7$_4$</td>
<td>300</td>
<td>0.091</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>ridge0.7$_5$</td>
<td>1000</td>
<td>0.050</td>
<td>42</td>
<td></td>
<td>ridge0.7$_5$</td>
<td>300</td>
<td>0.088</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>ridge0.7$_6$</td>
<td>1000</td>
<td>0.048</td>
<td>52</td>
<td></td>
<td>ridge0.7$_6$</td>
<td>300</td>
<td>0.086</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>ridge0.7$_7$</td>
<td>1000</td>
<td>0.046</td>
<td>61</td>
<td></td>
<td>ridge0.7$_7$</td>
<td>300</td>
<td>0.086</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td>ridge0.7$_8$</td>
<td>1000</td>
<td>0.043</td>
<td>73</td>
<td></td>
<td>ridge0.7$_8$</td>
<td>300</td>
<td>0.086</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td>ridge0.7$_9$</td>
<td>1000</td>
<td>0.041</td>
<td>84</td>
<td></td>
<td>ridge0.7$_9$</td>
<td>300</td>
<td>0.086</td>
<td>38</td>
</tr>
</tbody>
</table>

It can be seen that for each of the four output variables, both the MSPE$_{norm}$ improve when concatenated variables are used as input variables. For the QTYY1, QTYY2, and QTYY4, the best performing methods use about seven concatenated variables. In the case of QTYY3, the MSPEs$_{norm}$ continue decreasing when the amount of concatenated variables increases. In general it can be stated that using five or more concatenated variables significantly improves the performance of the regression method for each output variable.
The objective of this research was to examine whether the use of machine learning and other statistical techniques could be used to make earlier fault detection possible. The results of this report show that the application of ordinary least squares (OLS) and ridge regression methods to recurrently identify finite impulse response (FIR) models can construct accurate predictions of key variables in industrial processes. These methods can be used to extract information from a large set of historic data and to construct predictions. These predictions can be used for early fault detection in steady state processes, and can serve as a first step towards earlier fault detection in processes that do not necessarily operate at steady state. This conclusion is the result of the analysis of six questions. First: what kind of model can describe the dynamics of an industrial process? Second: can interpolation methods be used to fill in the missing output data and does this improve the available information in comparison to only using the samples for which output measurements are available and discarding the samples which only contain input measurements. Third: are on-line OLS methods able to match off-line first order principle based prediction methods? Fourth: can regression methods construct prediction methods with the same or improved accuracy, without using any prior knowledge about the system? Fifth: can these prediction methods be improved by using concatenated variables as inputs? Finally: are the identified methods able to reproduce prediction methods with the same accuracy for another process, without having to perform an entire analysis first? The analysis of each of these questions will be discussed individually in the chapter.

Using a FIR model to describe the dynamics of the process, means that when a multi-step ahead prediction is made, a single prediction is made: namely that of the multi-step ahead value. This is in contrast to, for instance, an autoregressive-moving average model with exogenous inputs (ARMAX) model, where multiple one-step ahead predictions are made to construct the final multi-step ahead prediction. Both models are used to construct prediction of key variables of the case study. The results show that the FIR actually performs better, in terms of MSPE$_{\text{norm}}$, than the ARMAX models. Therefore it is concluded that the dynamics of the process can be accurately described by a FIR model. It is possible that dynamic models perform differently when the prediction horizon is increased beyond an hour. Therefore re-
Concluding remarks

Evaluating the performance of different dynamic models for longer prediction horizons is an important next step in this research.

Another interesting type of models, that is worthwhile investigating, is artificial neural networks. In particular long short-term memories or other recurrent neural networks as these are able to deal with large prediction horizons. The downside is that it is highly unlikely that such methods could be generalised, which means that it would require to be completely reconfigured for each implementation. A lot more knowledge and effort would therefore be needed to calibrate and maintain the model. In this research, a brief analysis was performed to determine the potential of using artificial neural networks for the regression problem. However, with the limited time, no worthwhile results were found. This is probably more a reflection of the increased time and effort required to use artificial neural networks effectively for this application and not because this technique is unsuitable for use in this case. It might not be desirable to use artificial neural networks for every implementation, but for the key plants and processes, it may well be worth the extra time and effort. Therefore further research into the possibilities of using artificial neural networks for early fault detection is definitely required.

Three different interpolation methods were used to estimate the missing values of the key output variable. In addition a dataset was examined, which discarded the samples for which did not include an output value. Each of these four datasets were used to construct a prediction set. The dataset that discarded the samples with missing values, performed the worst in each experiment. The difference between the three interpolation methods was small. The zero-order hold methods performed slightly worse than the linear and spline method, and between the latter two, the differences were negligible. Therefore, it was concluded that a linear interpolation method should be used to fill in the missing output values.

A prediction method for the key output variable had been identified previous to this research. This method is a Wiener non-linear model, which uses four input variables and these input variables were determined by engineering principles. The model was identified off-line, but its intercept is corrected recursively using the error between estimate and measured value of the output. To verify whether an on-line identified model can achieve a similar performance as off-line model, OLS is used to identify a model on-line model with the input variables. The results show that the on-line identified model is able to predict the key output values, at least as well as off-line model does and it is therefore concluded that the on-line identified model can also accurately describe and predict the behaviour of the key output variable.

Five different regression methods were used to construct predictions about the key output variable. Because we wanted to construct predictions without using any prior knowledge of the system, each of the regression methods were applied to all the available variables as inputs for its models. Each regression method was applied to a dataset representing normal and abnormal operating behaviour. For both datasets it can be concluded, that the partial least squares (PLS) regression methods do not perform well. On the other hand each of the four other regression methods, OLS, PC regression, ridge regression, and least absolute shrinkage and selection operator (LASSO), were able to construct accurate predictions using all available variables as inputs. It can therefore be concluded that it is possible to construct predictions without using any prior knowledge about the system and it is even possible to construct more accurate predictions by including more input variables than the original four variables, which were determined using first principles. Each of the regression methods used
a large set of variables as inputs and were able to recognise which contained the relevant information. However, all of the variables were kept as inputs for the methods. A next step in this research is examine whether a number of the input variables could be excluded from the model. This could increase the robustness of the model and if the accuracy does not decrease significantly, this would certainly be a worthwhile exercise.

The results of the OLS, ridge regression, and LASSO methods were extremely similar and were better than any of the other results. Both the ridge regression and the LASSO method are a variation of the OLS method, where a regularisation term has been added in the estimation of the parameters. It could be the case that the results are so similar because this term doesn’t affect the estimated parameters much. However this was examined and it can be concluded that this is not the case. In both cases the extra term applies substantial regularisation to the estimates. A different consideration is the computation time of the different methods. As each methods has to identify a new model for each sample in the dataset, it is undesirable for the regression methods to be too intensive computationally. The LASSO identification is a quadratic optimisation problem and this means that its computation time is much longer than that of the OLS or ridge regression. It can be concluded that the ridge regression has the most desirable overall characteristics, as the performance of the three methods was so similar, the regularisation terms improve the robustness, and the LASSO is so intensive computationally.

Each of the regression methods uses an model to describe the dynamics of the process where a single output measurement is linked to a single vector of measurements, which contains the values of each of the input variables at a given time step. However, the use of concatenated variables as inputs of the FIR models is also examined. This means that a single output measurement is linked to multiple vectors of measurements, corresponding to the values of the input variables for consecutive time steps. The result of this is that the for both the datasets, the performance of the regression methods is improved. In addition the predicted output sets become less volatile. It is therefore concluded that the use of concatenated variables as inputs, can increase the robustness and improve the performance of the prediction methods.

The OLS and ridge regression method were also used to identify a FIR model of a different system as validation of their performance. This system is also a distillation column, but differs in number and type of inputs and outputs, as well as, in speed of the process, compared to the original system. Both methods were applied to construct predictions of each of the key variables and were able to produce accurate predictions. This performance was improved when concatenated variables were used as inputs. This validation is initially done for a single rolling window and prediction horizon, to investigate whether predictions can be made without any analysis. It is concluded that this is the case; however these predictions can be improved quite significantly by a simple analysis of which size of rolling window is optimal. This analysis does not require prior knowledge of the system and could very easily be automated into the procedure. A next step in this research is to verify the generality of the identified methods on other types of industrial processes.

The historic data that has been used in this research was available at a high frequency without the filtering that is generally applied to data stored in industrial plants. It is therefore likely that for further research available historical data would have a lower frequency and quality than the data that was used in this research. Therefore a next step is to evaluate whether the analysis is possible with this lower quality data or whether high quality data is a requirement. If the prediction methods were to be implemented in practice, this would
not be an issue, as the data available in the control system is of the same quality as that used in this research. In addition this research limited itself to examining process variables and modern industrial plants. However, modern industrial plants also have a large amount of diagnostic data available. So another next step of research is to investigate whether the addition of diagnostic data can also improve the performance of the fault detection methods.

The use of real industrial data in this research meant that it was not possible to select historic datasets e.g. datasets in which it was known that either no faults had occurred, or where a single specified fault had occurred. This was due to the fact that accurate logs faults and disturbances, both occurring in the process or induced by operators. Therefore datasets which closely represented both normal and ‘faulty’ operations had to be chosen. In the case of the faulty set, it was therefore not known whether an actual fault has occurred, and if it had, which type of fault. If datasets that have accurate fault logs are available, the automation of fault isolation and even diagnosis could be examined, making it easier to train and validate data-driven model-based fault detection methods. A next step in for this research is therefore to investigate exactly how useful these alarm logs currently are and what changes can be applied to make them a consistent source of information for automated process monitoring.

It has been concluded that implementation of the regression methods can be used to make accurate prediction of the key variables. In the case of a steady state variable, these predictions can be compared to some defined alarm-limit to determine whether a fault has occurred. However when a variable does not operate at steady state, this becomes more difficult. The predicted variable should be compared to some dynamic value in order to determine whether a fault has occurred. In the field of alarm management, a lot of research has been carried out in the field of alarm management, specifically in the area of dynamic alarm limit. Investigating the possibilities to incorporate these methods in order to more accurately detect any faults in the process could be a next step in this research. Another possibility would be to construct a second prediction corresponding to the predicted value of the key variable under normal operations. In the case that no fault is present in the system, this value should not differ from the other predicted value. However, in the case where a fault is present, the two predictions should differ. Therefore the problem with this method or these methods is that a double estimation error is added to the fault detection methods.
### A-1 Depentanizer

#### Table A-1: Description of the input variables in the C-X11

<table>
<thead>
<tr>
<th>Full Name</th>
<th>Name</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDX1:FCXX3.PV</td>
<td>FCXX3</td>
<td>MT/day</td>
<td>Inflow of C-X11</td>
</tr>
<tr>
<td>CDX1:FTXX1.PV</td>
<td>FTXX1</td>
<td>MT/day</td>
<td>Top reflux flow of C-X11</td>
</tr>
<tr>
<td>CDX1:FTXX2.PV</td>
<td>FTXX2</td>
<td>MT/day</td>
<td>Inflow of C-X12</td>
</tr>
<tr>
<td>CDX1:FTXX3.PV</td>
<td>FTXX3</td>
<td>MT/day</td>
<td>Flow of C-X21 to E-702</td>
</tr>
<tr>
<td>CDX1:FTXX4.PV</td>
<td>FTXX4</td>
<td>MT/day</td>
<td>Inflow of C-X13</td>
</tr>
<tr>
<td>CDX1:FTXX5.PV</td>
<td>FTXX5</td>
<td>MT/day</td>
<td>Full range tops drain</td>
</tr>
<tr>
<td>CDX1:FTXX6.PV</td>
<td>FTXX6</td>
<td>MT/day</td>
<td>Bypass flow of C-X12</td>
</tr>
<tr>
<td>CDX1:HCXX6.PV</td>
<td>HCXX6</td>
<td>%</td>
<td>Reflux ratio C-X11</td>
</tr>
<tr>
<td>CDX1:LTXX1.PV</td>
<td>LTXX1</td>
<td>%</td>
<td>Level in C-X11</td>
</tr>
<tr>
<td>CDX1:LTXX2.PV</td>
<td>LTXX2</td>
<td>%</td>
<td>Level in V-Y22</td>
</tr>
<tr>
<td>CDX1:LTXX7.PV</td>
<td>LTXX7</td>
<td>%</td>
<td>Level in C-X13</td>
</tr>
<tr>
<td>CDX1:LTXX3.PV</td>
<td>LTXX3</td>
<td>%</td>
<td>Level in V-Y21</td>
</tr>
<tr>
<td>CDX1:PTXX1.PV</td>
<td>PTXX1</td>
<td>barg</td>
<td>Pressure at tray 11 of the C-X11</td>
</tr>
<tr>
<td>CDX1:PTXX2.PV</td>
<td>PTXX2</td>
<td>barg</td>
<td>Pressure security at V-Y22</td>
</tr>
<tr>
<td>CDX1:QTXX2.PV</td>
<td>QTXX2</td>
<td>wt%</td>
<td>Mass fraction C4 in bottom C-X27</td>
</tr>
<tr>
<td>CDX1:TTX01.PV</td>
<td>TTX01</td>
<td>℃</td>
<td>Temperature at tray 11 of the C-X11</td>
</tr>
<tr>
<td>CDX1:TTX02.PV</td>
<td>TTX02</td>
<td>℃</td>
<td>Temperature of inflow to E-701</td>
</tr>
<tr>
<td>CDX1:TTX03.PV</td>
<td>TTX03</td>
<td>℃</td>
<td>Temperature at inflow to C-X11</td>
</tr>
<tr>
<td>CDX1:TTX04.PV</td>
<td>TTX04</td>
<td>℃</td>
<td>Temperature at top of the C-X11</td>
</tr>
<tr>
<td>CDX1:TTX05.PV</td>
<td>TTX05</td>
<td>℃</td>
<td>Temperature of flow E-Z72A/B</td>
</tr>
<tr>
<td>CDX1:TTX06.PV</td>
<td>TTX06</td>
<td>℃</td>
<td>Temperature of C-X11 inflow reboiler E-Z72B</td>
</tr>
<tr>
<td>CDX1:TTX07.PV</td>
<td>TTX07</td>
<td>℃</td>
<td>Temperature of C-X11 outflow reboiler E-Z72A</td>
</tr>
<tr>
<td>CDX1:TTX08.PV</td>
<td>TTX08</td>
<td>℃</td>
<td>Temperature of C-X11 outflow reboiler E-Z72B</td>
</tr>
<tr>
<td>CDX1:TTX10.PV</td>
<td>TTX10</td>
<td>℃</td>
<td>Temperature of inflow to C-X13</td>
</tr>
<tr>
<td>CDX1:TTX11.PV</td>
<td>TTX11</td>
<td>℃</td>
<td>Temperature of reflux flow C-X11</td>
</tr>
<tr>
<td>CDX1:TTX12.PV</td>
<td>TTX12</td>
<td>℃</td>
<td>Temperature of inflow to E-Z74</td>
</tr>
<tr>
<td>CDX1:TTX13.PV</td>
<td>TTX13</td>
<td>℃</td>
<td>Temperature of inflow to C-X12</td>
</tr>
<tr>
<td>CDX1:TTX14.PV</td>
<td>TTX14</td>
<td>℃</td>
<td>Temperature of full range tops drain</td>
</tr>
<tr>
<td>CDX1:TTX15.PV</td>
<td>TTX15</td>
<td>℃</td>
<td>Temperature of outflow cooling fluid E-Z14</td>
</tr>
<tr>
<td>CDX1:TTX16.PV</td>
<td>TTX16</td>
<td>℃</td>
<td>Temperature of inflow cooling fluid E-Z77</td>
</tr>
<tr>
<td>CDX1:TTX17.PV</td>
<td>TTX17</td>
<td>℃</td>
<td>Temperature of outflow cooling fluid E-Z78</td>
</tr>
<tr>
<td>CDX1:TTX18.PV</td>
<td>TTX18</td>
<td>℃</td>
<td>Temperature of inflow to E-Z70</td>
</tr>
<tr>
<td>CDX1:TTX19.PV</td>
<td>TTX19</td>
<td>℃</td>
<td>Temperature of inflow cooling fluid E-Z72A/B</td>
</tr>
<tr>
<td>CDX1:TTX20.PV</td>
<td>TTX20</td>
<td>℃</td>
<td>Temperature of outflow cooling fluid E-Z72A/B</td>
</tr>
<tr>
<td>CDX1:XCXX1.PV</td>
<td>XCXX1</td>
<td>MW</td>
<td>Energy transfer from C-X21 to E-Z72</td>
</tr>
</tbody>
</table>
### Table A-2: Description of the input variables in the C-X12

<table>
<thead>
<tr>
<th>Full Name</th>
<th>Name</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDX1:FCYY1.PV</td>
<td>FCYY1</td>
<td>MT/day</td>
<td>Inflow of C-X12</td>
</tr>
<tr>
<td>CDX1:FCYY2.PV</td>
<td>FCYY2</td>
<td>MT/day</td>
<td>Top reflux flow of C-X12</td>
</tr>
<tr>
<td>CDX1:FCYY3.PV</td>
<td>FCYY3</td>
<td>MT/day</td>
<td>Flow of iC5 drain</td>
</tr>
<tr>
<td>CDX1:FCYY4.PV</td>
<td>FCYY4</td>
<td>MT/day</td>
<td>Reboil flow of E-Z76A to C-X12</td>
</tr>
<tr>
<td>CDX1:FCYY5.PV</td>
<td>FCYY5</td>
<td>MT/day</td>
<td>Reboil flow of E-Z76B to C-X12</td>
</tr>
<tr>
<td>CDX1:FCYY6.PV</td>
<td>FCYY6</td>
<td>MT/day</td>
<td>Inflow of V-Z73</td>
</tr>
<tr>
<td>CDX1:FCYY7.PV</td>
<td>FCYY7</td>
<td>MT/day</td>
<td>Bypass flow of C-X12</td>
</tr>
<tr>
<td>CDX1:FTYY8.PV</td>
<td>FTYY8</td>
<td>MT/day</td>
<td>Flow of nC5 drain</td>
</tr>
<tr>
<td>CDX1:LCYY1.PV</td>
<td>LCYY1</td>
<td>%</td>
<td>Level in V-Y22</td>
</tr>
<tr>
<td>CDX1:LCYY2.PV</td>
<td>LCYY2</td>
<td>%</td>
<td>Level in C-X12</td>
</tr>
<tr>
<td>CDX1:LCYY3.PV</td>
<td>LCYY3</td>
<td>%</td>
<td>Level in V-Y23</td>
</tr>
<tr>
<td>CDX1:LTYY4.PV</td>
<td>LTYY4</td>
<td>%</td>
<td>Level in bottom C-X12</td>
</tr>
<tr>
<td>CDX1:PCYY1.PV</td>
<td>PCYY1</td>
<td>barg</td>
<td>Temperature at top of the C-X12</td>
</tr>
<tr>
<td>CDX1:TCY01.PV</td>
<td>TCY01</td>
<td>°C</td>
<td>Temperature at tray 57 of the C-X12</td>
</tr>
<tr>
<td>CDX1:TCY02.PV</td>
<td>TCY02</td>
<td>°C</td>
<td>Temperature of outflow heating fluid E-Z76B</td>
</tr>
<tr>
<td>CDX1:TYY03.PV</td>
<td>TYY03</td>
<td>°C</td>
<td>Temperature of C-X12 outflow reboiler E-706A</td>
</tr>
<tr>
<td>CDX1:TYY04.PV</td>
<td>TYY04</td>
<td>°C</td>
<td>Temperature of C-X12 outflow reboiler E-706B</td>
</tr>
<tr>
<td>CDX1:TYY05.PV</td>
<td>TYY05</td>
<td>°C</td>
<td>Temperature at inflow to C-X12</td>
</tr>
<tr>
<td>CDX1:TYY06.PV</td>
<td>TYY06</td>
<td>°C</td>
<td>Temperature of outflow E-Z12</td>
</tr>
<tr>
<td>CDX1:TYY07.PV</td>
<td>TYY07</td>
<td>°C</td>
<td>Temperature of outflow cooling fluid E-Z12</td>
</tr>
<tr>
<td>CDX1:TYY08.PV</td>
<td>TYY08</td>
<td>°C</td>
<td>Temperature of C-X12 inflow reboiler E-706A/B</td>
</tr>
<tr>
<td>CDX1:TYY09.PV</td>
<td>TYY09</td>
<td>°C</td>
<td>Temperature of top flow C-X12</td>
</tr>
<tr>
<td>CDX1:TYY10.PV</td>
<td>TYY10</td>
<td>°C</td>
<td>Temperature of top reflux flow of C-X12</td>
</tr>
<tr>
<td>CDX1:TYY11.PV</td>
<td>TYY11</td>
<td>°C</td>
<td>Temperature of inflow E-Z11</td>
</tr>
<tr>
<td>CDX1:TYY12.PV</td>
<td>TYY12</td>
<td>°C</td>
<td>Temperature of inflow heating fluid E-Z76A/B</td>
</tr>
<tr>
<td>CDX1:TYY13.PV</td>
<td>TYY13</td>
<td>°C</td>
<td>Temperature of outflow heating fluid E-Z76A</td>
</tr>
</tbody>
</table>
Appendix B

Model Analysis

B-1 Results of OLS method with four input variables

B-1-1 Interpolated output results

Figure B-1: Linear interpolated output results
Figure B-2: Zero-order hold interpolated output results

Figure B-3: Spline interpolated output results
B-1-2 Measured output results

Figure B-4: Measured output results. When a rolling window of 100 is taken only no concatenated variables can be applied, otherwise there would not be enough observation in the rolling window to perform a OLS.
**B-1-3 Comparison**

Figure B-5: Comparison of the output results for the three interpolation methods and the measured output set. Only the results of the models with no concatenated variables are shown.
B-2 Estimated parameters of OLS with four input variables

B-2-1 Rolling window size of 500 samples

Figure B-6: Estimates of \( \beta \) for the OLS with a rolling window of 500 and a prediction horizon of 20 for the period 10th through the 19th of December 2015
B-2-2 Rolling window size of 2000 samples

Figure B-7: Estimates of $\beta$ for the OLS with a rolling window of 2000 and a prediction horizon of 20 for the period 10th through the 19th of December 2015
Appendix C

Results
C-1 Normal operations dataset

C-1-1 Identified parameters using the OLS and ridge regression methods

Figure C-1: \( \hat{\beta} \) throughout the period 10th through 19th of December 2015 for the OLS and ridge regression models, with a rolling window of 1000 and a prediction horizon of 20
Figure C-1: Continued.
C-1-2 Results of regressions with concatenated variables

Figure C-2: The results of the regression methods using concatenated variables for the dataset spanning the period the 10th through the 19th of December 2015. The amount of concatenated variables of a regression method is indicated by its subscript.
C-2 Abnormal operations dataset

C-2-1 Amount of included predictors by the LASSO

Figure C-3: For each point in the dataset, the number of included variables is shown for the LASSO model that was constructed with a rolling window of 1000 and a prediction horizon of 20
C-2-2 Identified parameters using the OLS and ridge regression methods

Figure C-4: \( \hat{\beta} \) throughout the period 1st through 18th of January 2016 for the OLS and ridge regression models, with a rolling window of 1000 and a prediction horizon of 20
Figure C-4: Continued.
C-2-3 Results of regressions with concatenated variables

Figure C-5: The results of the regression methods using concatenated variables for the dataset spanning the period the 1st through the 17th of January 2016. The amount of concatenated variables of a regression method is indicated by its subscript.
Appendix D

Validation Results
D-1 Quality estimator datasets for the Deisopentanizer

Figure D-1: Data of the output variables of the C-X12
D-2 Results of regression methods of the Deisopentanizer

Figure D-2: QTYY1
Validation Results

Figure D-3: QTYY2

(a) RW = 300
(b) RW = 500
(c) RW = 1000
(d) RW = 2000

Figure D-4: QTYY3

(a) RW = 300
(b) RW = 500
(c) RW = 1000
(d) RW = 2000
D-2 Results of regression methods of the Deisopentanizer

Figure D-5: QTYY4


### List of Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>APC</td>
<td>Advanced Process Control</td>
</tr>
<tr>
<td>ARC</td>
<td>ARC Advisory Group</td>
</tr>
<tr>
<td>BLC</td>
<td>Base Layer Control</td>
</tr>
<tr>
<td>NUI</td>
<td>normally unmanned installation</td>
</tr>
<tr>
<td>DCS</td>
<td>distributed control system</td>
</tr>
<tr>
<td>CDU</td>
<td>crude distiller unit</td>
</tr>
<tr>
<td>RQE</td>
<td>Robust Quality Estimator Pro</td>
</tr>
<tr>
<td>ECP</td>
<td>effective cut point</td>
</tr>
<tr>
<td>SI</td>
<td>separation index</td>
</tr>
<tr>
<td>IBP</td>
<td>initial boiling point</td>
</tr>
<tr>
<td>EP</td>
<td>end point</td>
</tr>
<tr>
<td>CDX1</td>
<td>Crude Distiller X1</td>
</tr>
<tr>
<td>CDX2</td>
<td>Crude Distiller X2</td>
</tr>
<tr>
<td>C-X11</td>
<td>Depentanizer</td>
</tr>
<tr>
<td>C-X12</td>
<td>Deisopentanizer</td>
</tr>
<tr>
<td>C-X13</td>
<td>Deisohexanizer</td>
</tr>
<tr>
<td>C-X21</td>
<td>Crude Splitter 1</td>
</tr>
<tr>
<td>C-X22</td>
<td>Crude Splitter 2</td>
</tr>
<tr>
<td>C-X27</td>
<td>Debutanizer</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>C-X28</td>
<td>Gasoline Splitter</td>
</tr>
<tr>
<td>NOD</td>
<td>normal operations dataset</td>
</tr>
<tr>
<td>AOD</td>
<td>abnormal operations dataset</td>
</tr>
<tr>
<td>SOD</td>
<td>standard operations dataset</td>
</tr>
<tr>
<td>QT1</td>
<td>QTXX1</td>
</tr>
<tr>
<td>QE1</td>
<td>QEXX1</td>
</tr>
<tr>
<td>C5</td>
<td>pentane</td>
</tr>
<tr>
<td>nC5</td>
<td>n-pentane</td>
</tr>
<tr>
<td>iC5</td>
<td>isopentane</td>
</tr>
<tr>
<td>cC5</td>
<td>cyclopentane</td>
</tr>
<tr>
<td>nC4</td>
<td>n-butane</td>
</tr>
<tr>
<td>UCL</td>
<td>upper control limit</td>
</tr>
<tr>
<td>LCL</td>
<td>lower control limit</td>
</tr>
<tr>
<td>MSPE</td>
<td>mean squared prediction error</td>
</tr>
<tr>
<td>FIR</td>
<td>finite impulse response</td>
</tr>
<tr>
<td>ARMAX</td>
<td>autoregressive-moving average model with exogenous inputs</td>
</tr>
<tr>
<td>MISO</td>
<td>multiple inputs, single output</td>
</tr>
<tr>
<td>OLS</td>
<td>ordinary least squares</td>
</tr>
<tr>
<td>LASSO</td>
<td>least absolute shrinkage and selection operator</td>
</tr>
<tr>
<td>PLS</td>
<td>partial least squares</td>
</tr>
<tr>
<td>PLSC</td>
<td>partial least squares component</td>
</tr>
<tr>
<td>PCA</td>
<td>principal components analysis</td>
</tr>
<tr>
<td>PC</td>
<td>principal component</td>
</tr>
<tr>
<td>SVD</td>
<td>singular value decomposition</td>
</tr>
<tr>
<td>EM</td>
<td>expectation maximization</td>
</tr>
</tbody>
</table>