Novel Anomaly Detection Algorithms for Major and Critical Items of Plant

Andrew D. Kenyon

A thesis submitted for the degree of Doctor of Philosophy to
Institute for Energy and Environment
University of Strathclyde

2013

This thesis is the result of the authors original research. It has been composed by the author and has not been previously submitted for examination which has led to the award of a degree.

Signed:

Date:

Abstract

For an electrical utility, the unplanned outage of a major plant item can lead to a substantial loss of revenue. In addition, outage costs must take into account the cost of any required repairs, or even replacements. It is therefore best to avoid long outages through a combination of thorough maintenance and comprehensive condition monitoring.

The condition monitoring techniques available to a utility are related to the quality and quantity of data present. In order to capture the dynamics of complex systems, high frequency data is preferred. Lack of a sufficiently high sampling rate can render traditional methods of modelling the dynamics inaccurate or impractical.

This thesis proposes an adaptation of machine learning algorithms to model major and critical plant items. Artificial neural networks (ANNs) are shown to allow anomaly detection in pulverising mills and gas turbines (GTs). Hidden Markov models (HMMs) are shown to allow the modelling of GT dynamics despite the absence of high frequency data, and provide tracking of machine health degradation. In addition, a multi-agent system (MAS) is proposed that incorporates several models based on subcomponents of the turbine. The use of MAS technology also allows other algorithms to be easily added to the system, to provide additional condition monitoring capability. Finally, the use of evidence combination allows the agents of different sub-components and differing algorithms to work together to improve condition monitoring performance.

Acknowledgements

I would like to thank Prof. Stephen McArthur for giving me the opportunity to carry out my research as part of the Intelligent Systems Team. I would also like to thank Dr. Victoria Catterson for her invaluable support and advice over the course of my PhD.

Thanks are also due to SSE for providing not only the financial means to conduct this research, but also for their contributions of technical expertise, domain knowledge and experimental data. Particular thanks are due to Dr. John Twiddle and Dr. Seth Muthuraman.

I would like to thank my close friends and family for their endless words of support. A special thanks must go to my parents, without whom I might not have found the drive to finish this undertaking.

Finally, I would like to dedicate this work to the late Barbara and William Grant. To my Aunt Barbara and Uncle Bill, I hope that this is a fitting tribute, and a representation of how the love you showed me has shaped the man I am today.

Contents

List of Figures

List of Tables

Glossary of Abbreviations

ACL	agent communication language.
AI	artificial intelligence.
AMS	Agent Management System.
ANN	artificial neural network.
AOP	agent-orientated programming.
BBN	Bayesian belief network.
BDI	belief, desire, intention.
BMU	best matching unit.
CA-EN	CAusal ENgine.
CBR	case-based reasoning.
CCGT	combined cycle gas turbine.
CIM	Common Information Model.
COMMAS	COndition Monitoring Multi-Agent System.
CSV	comma separated value.
DBSCAN	density-based spatial clustering of applications with noise.
DF	Directory Facilitator.
DFR	digital fault recorder.
DHMM	discrete HMM.
DP	Differential Pressure.
DVC	differential voltage control.
EM	expectation maximisation.

FFT	fast Fourier transform.
FIPA	Foundation for Intelligent Physical Agents.
FIPA-ACL	FIPA Agent Communication Language.
FIPA-SL	FIPA Semantic Language.
GMM	Gaussian mixture model.
GT	gas turbine.
GUI	graphical user interface.
HMM	hidden Markov model.
HP	high pressure.
HRSG	heat recovery steam generator.
ICA	independent component analysis.
IEEE	Institute of Electrical and Electronics Engineers.
IGV	inlet guide vane.
IP	intermediate pressure.
IxTeT	Indexed Time Tables.
JADE	Java Agent DEvelopment framework.
JVM	Java Virtual Machine.
KADS	Knowledge Acquisition and Documentation Structuring.
KBS	knowledge-based system.
KQML	Knowledge Query and Manipulation Language.
LL	log-likelihood.
LP	low pressure.
MAS	multi-agent system.

MBR	model-based reasoning.
MHMM	mixture HMM.
MIKE	Model-based and Incremental Knowledge Engineering.
MIMO	multiple input-multiple output.
MLP	multi-layer perceptron.
MTS	Message Transport System.
NOx	Nitrogen Oxides.
OEM	Original Equipment Manufacturer.
OOP	object-orientated programming.
PA	primary air.
PCA	principal component analysis.
PD	partial discharge.
PDF	probability density function.
PEDA	Protection Engineering Diagnostic Agents.
PI	plant information.
QDE	qualitative differential equation.
RBES	rule-based expert system.
RBF	radial basis function.
RCA	root cause analysis.
RMA	Remote Monitoring Agent.
SBM	similarity-based modelling.
SCADA	supervisory control and data acquisition.
SDF	symbolic dynamic filtering.
SISO	single input-single output.
SOM	self-organising map.
SSE	Scottish and Southern Energy.

TC	Thermocouple.
UML	Universal Modelling Language.
WEKA	Waikato Environment for Knowledge Analysis.
WFF	well-formed formula.

Chapter 1

Introduction

1.1 Introduction to the research

"The art of condition monitoring should be minimalist, to take the minimum measurements necessary and by analysis [allow fault] detection and diagnosis of the machine. A condition can then be inferred, in minimum time, giving a clear indication of incipient failure modes." [1]

The above quote defines both the purpose of condition monitoring, and the requirements for the methods employed. Condition monitoring must represent the state of the monitored subject, and do so in a timely and meaningful manner. This introduction provides a high-level review of condition monitoring approaches, with greater detail provided in later chapters.

Condition Monitoring has three possible goals:

Anomaly Detection

Otherwise known as novelty detection. The process of monitoring parameters related to the subject, and if they differ sufficiently from a perceived notion of normality, the subject is considered to have entered an anomalous state and an alarm may be raised [2]. In this type of condition monitoring there is no diagnosis of the nature of the fault, only that the monitored parameters have deviated from normal values [3].

Fault Diagnosis

Attempting to diagnose a problem by analysing data and determining the particular fault which has occurred [3]. Often, fault diagnosis will not determine that

there is a fault; only indicate the most likely fault that may have occurred.

Prognosis

Predicting, based on various factors relating to the health of the asset, the time until failure, or future health of that asset [4]. This may then be used to inform a maintenance strategy.

For the most part, this thesis will consider the task of detecting and diagnosing an existing fault.

Condition monitoring systems may provide one or all of these capabilities. There are examples of systems that combine anomaly detection and fault diagnosis [5] [6]. In the cited cases an anomaly detection algorithm is used first to detect whether a fault has occurred and if so, delegate the task of identifying the particular fault to a fault diagnosis algorithm. Some algorithms may be a mixture of the two approaches: an example would be a state-transition algorithm that represents several normal and anomalous states, allowing the system to determine the presence and type of a fault based on the current state.

For large and expensive items of plant, whose operation is critical to that of the plant, condition monitoring is especially relevant. The ability to consistently perform as designed is vital to their commercial viability. The degradation in performance of major and critical generation assets can lead to reduction in output that directly correlates to a loss of revenue. Performance loss can be for a number of reasons, and for known reasons can be managed with proper maintenance [7]. An example would be compressor fouling in gas turbines, which can be controlled with regular water washes. Nevertheless, the ability to monitor variations in performance is vital to a utility to allow it to optimise its maintenance schedule and increase the reliability and cost effectiveness of its assets [8].

In addition to expected degradation from known phenomenon, it is possible that an underlying fault could be present. This may or may not lead to a cumulative loss in performance. Such a fault may therefore be difficult to detect and can result in serious damage to the asset if not prudently corrected [9]. The critical failure of a major and critical item of plant could result in a long, costly, outage. Expensive repairs or even a replacement may be required before normal operation can resume. While such a fault may not obviously affect asset performance in the lead-up to failure, it will affect the underlying behaviour on some level. Detailed, timely and accurate condition monitor-

ing and diagnostic support for generation assets is important to successfully operate a power plant.

This research has been carried out in conjunction with Scottish and Southern Energy (SSE) plc, with the aim of improving and expanding the existing condition monitoring capabilities for major and critical items of plant. This is made possible by the application of artificial intelligence (AI)-based techniques. This thesis reviews several techniques for the condition monitoring of major and critical plant items, with particular focus on pulverising mills and gas turbines.

Pulverising mills must run continuously to provide fuel for boilers of large coal-fired plants. Gas turbines (GTs) are used in several ways, such as for peaking power or for cold-starts in large units. However, when they are used in combined cycle gas turbine (CCGT) units, they too are required to run for long periods of time where they are used for base load. Long periods of continuous operation limits the down-time for manual inspection and testing, while increasing the wear on the asset. Therefore, an on-line system that can detect problems while these systems are operating is a valuable commodity.

As well as reviewing several techniques this thesis successfully applies an approach most appropriate to each application. For pulverising mills, the use of artificial neural networks (ANNs) is demonstrated to allow the detection of primary air faults where more conventional techniques have proven unsuccessful. For GTs, both ANNs and hidden Markov models (HMMs) are shown to be effective, with mixture HMMs (MHMMs) proving to be more capable of fault detection and degradation tracking.

This thesis proposes a condition monitoring system to allow the monitoring of gas turbines for power generation. The system uses HMMs to detect variations in underlying gas turbine behaviour. The system makes use of multiple models, each modelling a different sub-component, to provide an anomaly detection capability. The implementation of evidence combination techniques across models and algorithms facilitates fault isolation and diagnosis. The architecture of the system, utilising a multi-agent system (MAS) approach, ensures a flexible and future proof solution able to incorporate additional algorithms as required.

1.2 Justification for research

The quality and quantity of data available for monitoring asset behaviour can vary between utility, plant, and even unit. Often, there will be a plant-wide database containing data from every plant item. The use of this data can allow the training of machine learning algorithms which in turn allows the detection of faults that may otherwise be missed. For this reason, this thesis reviews the field of AI to develop condition monitoring algorithms capable of detecting faults in major and critical plant items.

Pulverising mills in operation in large power-plants may contain a condition monitoring system based on trending of variables. However, such a system may not be sufficient, and a more sophisticated approach may be necessary and desired. Mathematical models for pulverising mills are extremely difficult to develop successfully due to the complexity and non-linearity of the system [10] [11]. Variation between designs also means that existing models may not be adaptable. A knowledge-based approach may be possible, but a utility will often not possess sufficient expertise to make this possible, and the variation and complexity would require a separate knowledge base for at least each type of mill.

This thesis proposes data-based machine learning techniques for the monitoring of pulverising mills, removing the need for detailed mathematical models or lengthy knowledge elicitation. ANNs were chosen due to their ability to model non-linear systems [12] such as pulverising mills. This approach is successfully tested on a ball and race pulverising mill, where existing applications of ANNs have focused on ball mills. An existing condition monitoring system was unable to detect a pending pressure excursion in the primary air system, that ultimately led to a major failure. The proposed system is shown to be capable of detecting an anomaly and raising an alarm in advance of the failure.

Monitoring of Gas Turbines in CCGT power stations represents several challenges as the information available from the plant database is generally considered insufficient to completely model the dynamics of a gas turbine, due to a low sampling rate. To address this, it is common to augment this with some form of turbine specific monitoring software, often dedicated to frequency analysis of shaft vibration. Such a system is not always available and requires high-frequency vibration data. In addition, the reliance on a relatively small variable set and one approach means it is only able to detect a subset of faults.

The use of probabilistic models, in the form of HMMs is proposed to allow the mod-

elling of the dynamics of a gas turbine. This thesis aims to show that such a model does not necessarily require high-frequency data, and compares favourably to other behaviour monitoring algorithms. To provide sufficient detail while still monitoring the behaviour of the whole turbine, it is necessary to use several models. This also provides a level of fault isolation, and can assist with diagnoses of a likely cause of any anomalous behaviour.

The HMMs are shown to reflect changes in GT behaviour. However the interpretation of the log-likelihood (LL) output of the models is a potentially time consuming process. This is compounded by the use of multiple models. To address this, this thesis provides a comparison of several metrics that may allow this process to be automated. A process of linear regression is also applied to the output of the HMMs to allow the degradation in machine condition over time to be identified.

While probabilistic models are shown to be effective, no single algorithm is optimal for all faults and conditions. For this reason a multi-agent architecture is proposed for all deployed models. While there are existing systems which incorporate several models [13], and even allow new models or knowledge to be added, they cannot incorporate additional algorithms without a recompilation of source code and a restart of the system. Typically, additional algorithms may not be provided until the next software version by the system provider. The use of a multi-agent system, as well as the choice of an open source agent platform and standardised communication protocols ensures maximum flexibility in the deployment options of the system. This allows additional algorithms to be deployed on the system with no interruption, providing a great deal of flexibility in the algorithms used.

The use of multiple models necessitates some way of corroborating between the models, in order to determine the validity, nature and cause of any apparently anomalous behaviour. Where the system incorporates additional algorithms to the probabilistic models, further corroboration is required. This thesis proposes a rule-based evidence combination technique to allow the best possible diagnosis in any particular configuration.

The following items are considered novel contributions to research in this field:

- Modelling of the pressures in a ball and race pulverising mill, without the use of complex mathematical models, through the application of ANNs.
- Probabilistic modelling of GTs using low-frequency data.

- An extensible multi-agent architecture incorporating several models and algorithms.
- Comparison of several metrics for automated fault detection based on analysis of HMM log-likelihood results, including new approaches based on sequence variance and rate-of-change.
- The application of linear regression to allow tracking of the degradation of performance in gas turbines.
- Rule-based evidence combination techniques to allow corroboration between diagnostic agents.

1.3 Thesis overview

This thesis comprises 8 chapters, including this introduction. Chapter 2 provides a review of several artificial intelligence techniques, including those ultimately implemented as part of the multi-agent system.

Chapter 3 provides a summary of MASs, along with some of the particular enabling technologies necessary for the proposed condition monitoring system and examples of MASs successfully applied to industrial applications.

Chapter 4 reviews the approaches typically taken to condition monitoring of pulverising mills. The chapter continues with the successful application of ANNs to an existing pulverising mill. The chapter concludes with an example of the successful detection of an anomaly related to the primary air fan venturi, where more conventional and pre-existing approaches were unable to detect the fault before damage and asset failure occurred.

Chapter 5 provides a brief summary of the main components and principles of a gas turbine followed by an overview of gas turbine monitoring in the power industry. Further sections document the application of several techniques which were applied in an attempt to successfully monitor the dynamics of a GT despite a low sampling rate. The chapter concludes with a demonstration of a successful HMM-based approach, and provides promising results when compared to an existing system using industrial data.

Chapter 6 includes a detailed description of the proposed condition monitoring multi-agent system. This includes a description of the various agents, communication between the agents, and the behaviour of the system when presented with real GT data.

The multi-agent system contains more than 100 agents per GT, capable of anomaly detection across the turbine.

Chapter 7 includes a section on the automated approaches to interpret the results from the HMMs agents. Further sections present a rule-based approach to evidence combination. This allows fault diagnosis to be performed based on the HMM and thresholding agents. A case study is then provided to further illustrate the capabilities of the algorithms, evidence combination techniques and the agent system as a whole when presented with industrial data.

Chapter 8 concludes with a summary of the work presented in this thesis and areas for further research in the future.

1.4 Associated publications

Publications which have arisen as result of the work described in this thesis are as follows:

- A. D. Kenyon, V. M. Catterson, S. D. J. McArthur, and J. Twiddle, "Application of Hidden Markov Models to Gas Turbine Anomaly Detection," *Accepted, IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*

- A. D. Kenyon, V. M. Catterson, and S. D. J. McArthur, "Development of an Intelligent System for Detection of Exhaust Gas Temperature Anomalies in Gas Turbines," *Insight: The Journal of the British Institute of Non-Destructive Testing*, vol. 52, no. 8, 2010

- A. D. Kenyon, V. M. Catterson, and S. D. J. McArthur, "Development of an Intelligent System for Detection of Exhaust Gas Temperature Anomalies in Gas Turbines," *CM 2010 and MFPT 2010 : The Seventh International Conference on Condition Monitoring and Machinery Failure Prevention Technologies*, vol. 52, no. 8, pp. 419–423, 2010

- A. D. Kenyon, V. M. Catterson, S. D. J. McArthur, and J. Twiddle, "Design of an Intelligent Diagnostic Architecture to Support the Condition Monitoring of Power Generation Assets," in *Proceedings of the 44th International Universities Power Engineering Conference (UPEC)*, 2009

Chapter 2

Intelligent System Techniques

The following chapter reviews a number of techniques from the field of AI. The first section describes the data mining methodology applied in the selection of intelligent algorithms for condition monitoring. In the subsequent sections, some of the intelligent system techniques applicable to the condition monitoring of plant assets are reviewed.

2.1 Data Mining Methodology

In order to make sense of the large amount of data available for plant items, a methodology is necessary, through which a process of data mining is possible. While there are differing data mining methodologies [18] [19] [20], and a number of potential definitions of data mining, in this thesis data mining is considered to be "non-trivial extraction of implicit, previously unknown, and potentially useful information from data" [19]. The chosen structure for data mining [20] consists of a number of steps, outlined below. On first pass, they are normally performed in the order introduced here, but data mining is an iterative process. The two main phases, comprising preparation of the data (steps 2.1.1-2.1.3), and technique selection and testing (2.1.4-2.1.5), are usually repeated several times until a satisfactory result is reached.

2.1.1 Familiarisation

This stage encompasses the initial discussions with the data provider, including domain experts and those already familiar with comparable data. This will also include reading any literature or event reports relevant to the data. The amount, scope and nature of the

data must be determined. The aim of this stage is to improve general understanding and knowledge of the data and how it applies to the problem domain.

2.1.2 Visualisation

Visualisation includes analysis of the data at a relatively high level. This is achieved primarily through graphical representations of the data. This stage involves attempting to identify any visual inconsistency or pattern in the data made apparent through the use of different techniques. These techniques can include trending, clustering or transformations on the data.

Dimensionality reduction is often performed at this stage to allow high dimensional data to be displayed in two or three dimensional diagrams and to allow patterns to be more easily identified visually.

2.1.3 Data Reduction and Cleaning

After familiarisation with the data and the identification of any patterns through visualisation, it may be possible to reduce the amount of data being considered. The aim of this stage is to remove data unnecessary to identify the state or condition of the monitored asset. This may include removing redundant or irrelevant data, especially in multivariate data where some readings may be strongly correlated or where variables are not related to the relevant outcome being considered. Dimensionality reduction also occurs at this stage, but to decrease the scope of the problem and increase the choice and effectiveness of available techniques, rather than to better visualise the data.

Some examples of the kind of processes used at this stage include independent component analysis (ICA), where it may be possible to identify the independent components that make up a multivariate signal. This would be in the hope that a number of independent components capable of representing the state of the system could be extracted from a larger number of parameters. Another example is clustering, which may allow the behaviour of the system to be represented as a series of state transitions. This may be more efficient and revealing than attempting to interpret the potentially high-dimensional data directly.

Following this stage it is common to re-familiarise oneself with the now cleaned and reduced dataset, before once again repeating the process. It may take several iterations before a dataset is considered appropriate for a diagnostic technique to be selected.

2.1.4 Technique Selection

After performing the preceding steps a sufficient number of times to identify features within the data, and reducing the dimensionality of the data as much as possible, the potential techniques for anomaly detection or fault diagnosis must be considered. The exact methods considered depend on the data (real or complex, linear or non-linear), the application (anomaly detection, fault diagnosis, or both) and the requirements (accuracy, particular requirements such as a low number of fault positives, single outcome or a vector of possibilities).

There may be a number of possibilities, but a single technique that is considered to best fit the data and the requirements will be selected and the process will proceed to implementation and testing.

2.1.5 Implementation and Testing

The selected technique must be implemented, which includes the training stage for machine learning algorithms. The technique must then be tested on fresh data to ensure that it is applicable to data other than that it has seen (or trained on) before. If the technique proves unsatisfactory the process will revert to technique selection, before proceeding again to implementation and testing. This cycle will repeat until a satisfactory technique has been identified and verified.

2.2 Types of Algorithms

When applied to condition monitoring problems, intelligent system techniques can be considered to fall under one of the following three categories:

Data-based

Techniques capable of deriving relationships between variables based on sample data, and "learning" the behaviour of a system. Once trained, they can recognise a deviation from normal behaviour or the presence of a specific type of fault. This approach is useful where the relationships between parameters are not fully understood. These techniques require sufficient data to train and test any model.

Knowledge-based

Approaches based on capturing pre-existing knowledge of a system. This may

involve knowledge elicitation in order to capture implicit expert knowledge. This allows automation of the analysis an engineer would perform in a knowledge-based system (KBS). Such an approach requires that sufficient knowledge of the system in question is available.

Model-based

Based on a model of the system, this approach compares the actual and modelled values of variables to determine the presence of a fault. Techniques such as model-based reasoning (MBR) can be used to determine the components most likely to be responsible for abnormal behaviour [21]. This requires a first-principles understanding of the operation of the system.

The distinction between these areas is not fixed, and some techniques may be considered to fit into more than one. For example, a model-based reasoning approach could use a data-based machine learning technique, such as neural networks, to represent the sub-components of a system, while still utilising a higher level model. Also, as model-based techniques require substantial knowledge of the system, they are sometimes considered a sub-category of knowledge-based techniques.

The techniques described in the following sections all fall under the data-based or knowledge-based categories.

2.3 Principal Component Analysis

Principal component analysis (PCA) [22] is the process of extracting, from a number of interrelated and correlated variables, a new set of uncorrelated variables called the principal components [23]. The process involves transforming the data so that the first component has the maximum variance possible. Each subsequent component is calculated in a similar manner, with the additional requirement that the component is uncorrelated with any preceding components, i.e. is orthogonal. The principal components are ordered by variance. Where the original data is strongly correlated, the first few components contain most of the variation in the data, and the higher-numbered components can be disregarded with little loss of information [24].

PCA can be used for dimensionality reduction and data compression. It is also useful in establishing and visualising relationships between variables.

2.4 Independent Component Analysis

ICA is a signal processing method to extract independent components given observed data from many unknown sources [25]. The components may be across several of these sources, and require separation.

Several pre-processing steps are necessary, including centring, whitening, and possibly dimensionality-reduction [26]. After pre-processing, ICA uses the principles of statistical independence to identify and separate the components, while measuring Gaussianity to distinguish these components from noise.

ICA is in some ways an extension of PCA [27]. However, while principal components must be orthogonal, this requirement does not apply to the independent components of ICA. Also, while attempting to maximise the variance for a component PCA will potentially include aspects of several correlated sources, which may or may not be independent. Conversely, ICA will try to separate out activity into as many independent components as possible, based on the number of statistically independent non-Gaussian signals. Therefore, ICA is the more appropriate choice for a blind source separation problem: separating out the voices of several people in a conversation picked up as combined signals from a number of microphones [25]. A simple example of such a problem is provided below.
Taking a two source/two sensor problem, we can define the inputs to the sensor as:

$$x_1(t) = a_{11}s_1 + a_{12}s_2 \tag{2.1}$$

$$x_2(t) = a_{21}s_1 + a_{22}s_2 \tag{2.2}$$

The equations 2.1 and 2.2 give the equation for the signals picked up by the sensors, where s_1 and s_2 are original signals, $x_1(t)$ and $x_2(t)$ are the signals observed by the sensors, and a_{ij} is the relative strength of the original signal components as perceived by the sensors [26].

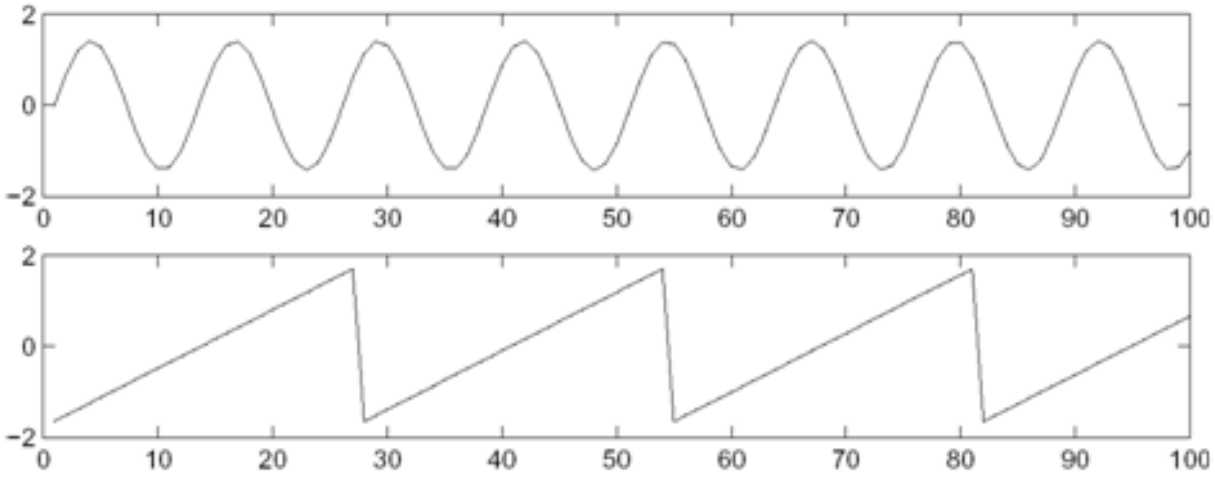

Figure 2.1: Two Non-Gaussian Signals [26].

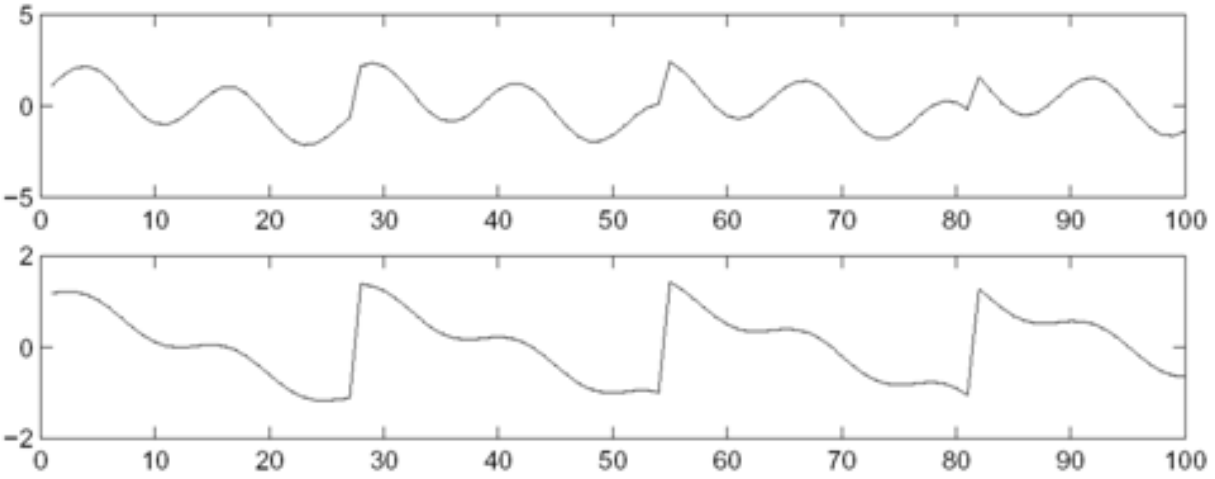

Figure 2.2: The signals, now mixed, observed from two sensors [26].

The above Figures show the original signals (Figure 2.1), and the mixed signals (Figure 2.2). The use of ICA allows the original signals to be reconstructed (Figure 2.3).

ICA may be used for condition monitoring, as it can allow the independent components of a multivariate signal to be identified. This may allow an underlying component, which may indicate the presence of a fault or act as a precursor to a failure, to be identified where it may otherwise be seen as unrelated noise across a number of sensor readings. ICA has already been applied successfully to gas turbines [28], and was able to detect faults which other techniques, such as PCA [22] alone, could not. It may also be used as a form of dimensionality reduction, where the independent components may

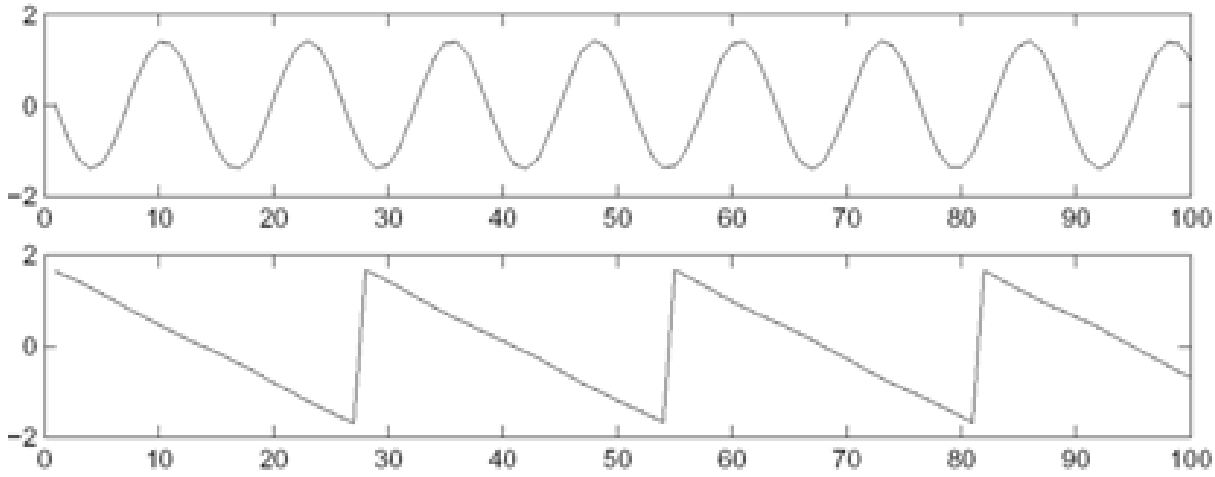

Figure 2.3: The Independent Components [26]. Note the close correlation to the original signals, albeit with changed signs.

be lower in number than the number of sensors.

2.5 Clustering

Clustering can have a number of uses both as a condition monitoring algorithm in its own right, and in the data mining process. Clustering is similar to classification with supervised learning algorithms, such as ANNs [29] and support vector machines (SVMs) [30], in that the data is attributed to one of the identified clusters or classes. However, clustering algorithms are able to determine the clusters without labelled data, instead identifying the clusters based on patterns in the data, and as such are unsupervised approaches. Clustering can be used to visualise the data, or the clusters may be identified as states which a signal transitions between.

Clustering algorithms are particularly diverse; for example, there are over 20 different clustering algorithms in the WEKA toolkit alone [31]. Each has different caveats in their use, using different parameters, sometimes requiring a priori knowledge [32]. The two clustering techniques used in this thesis are K-means, and density-based spatial clustering of applications with noise (DBSCAN).

2.5.1 K-Means

K-means is often one of the first algorithms used because it is well-known and relatively simple. In K-Means [33] the user supplies the number of clusters "K" that the algorithm will divide the data into. The first step of K-Means is randomly generating the cluster centres. The data points are then assigned to the nearest cluster and the centre recalculated. This process is iterated, and when the cluster assignments remain static, meaning that further recalculations will have no effect, the algorithm is finished.

While K-means is a proven technique, it has some disadvantages. The most obvious is the requirement to specify the number of clusters in advance. There are variants, such as X-means, that calculate the number of clusters statistically during pre-processing [32], but this is only an estimate. Another disadvantage is that because the initial cluster centres are calculated at random, the technique may give different results each time it is used. These disadvantages can be somewhat mitigated by thorough testing of the data. The algorithm can be run multiple times over a range of "K" values until the result is deemed satisfactory.

2.5.2 DBSCAN

DBSCAN is a clustering algorithm [34] based on the principle of ϵ-neighbourhood. The cluster begins with a data point, and then searches for other data points within ϵ distance. Points within this distance are added to the cluster, and the process continues with the newly added points searching for neighbours, and so on.

One of DBSCAN's strengths is that it does not require the number of clusters to be known a priori. However, it does require the ϵ value and the minimum number of points for a cluster to be set. The combination of the two variables has a direct effect on the result, and are perhaps less intuitive to define than the number of clusters of K-means.

It is possible for points to be unallocated, and remain outside all clusters, which may be a strength or weakness depending on the application. For example, it may reduce noise in particularly noisy data. Conversely, where classification is important an unclassified data point may be counter-productive.

2.6 Artificial Neural Networks

An ANN [29] is a type of machine learning algorithm. ANNs are made up of several nodes that mimic the firing and interconnection of neurons in the brain. Each link between nodes has a weight applied to it, and by adjusting this, the output of the node will change. The interconnections between nodes and the subsequent manipulation of the weights of the nodes allow the network learn. All ANNs have two modes - training, where the network learns the relationships between inputs and output, and testing, where the network estimates the output for a set of given inputs. The network's success during testing is related to how well it has learned the relationships during training.

ANNs are particularly useful for their ability to learn non-linear relationships [12]. This is made possible by the use of a non-linear activation function, which allows a non-linear relationship between the input and output. Two types of ANN were utilised in the work presented in this thesis, and are discussed the following sections.

2.6.1 Multi-Layer Perceptron

A multi-layer perceptron (MLP) is a feed-forward ANN consisting of multiple layers. It is feed-forward because, after training is complete, values only pass forward between the layers. This will typically consist of an input layer, one or more hidden layers, and an output layer [35]. The hidden layer is so called because the inputs and outputs of these nodes are hidden from the outside.

Each node of a layer is connected to every node of the following layer. Each link has a weight associated with it, with the node at the end of the links calculating a weighted sum for its inputs. The node then applies the input to a transfer function, producing the node's output. The transfer function is usually non-linear, with a sigmoid function being a common example. It is this non-linear function that enables the MLP to model non-linear systems [12]. A single hidden layer is sufficient for any approximation, however, multiple hidden layers may reduce the number of necessary nodes or connections [36]. Figure 2.4 shows a simple MLP with a single hidden layer.

The most common training algorithm for MLPs is back-propagation [37]. This involves presenting the network with data for which it must attempt to learn the relationships between inputs and output. During training, the error between the MLP output and the actual value is propagated backwards, through the links between nodes, and adjustments are made to the weight values in an attempt to minimise the error [35]. The

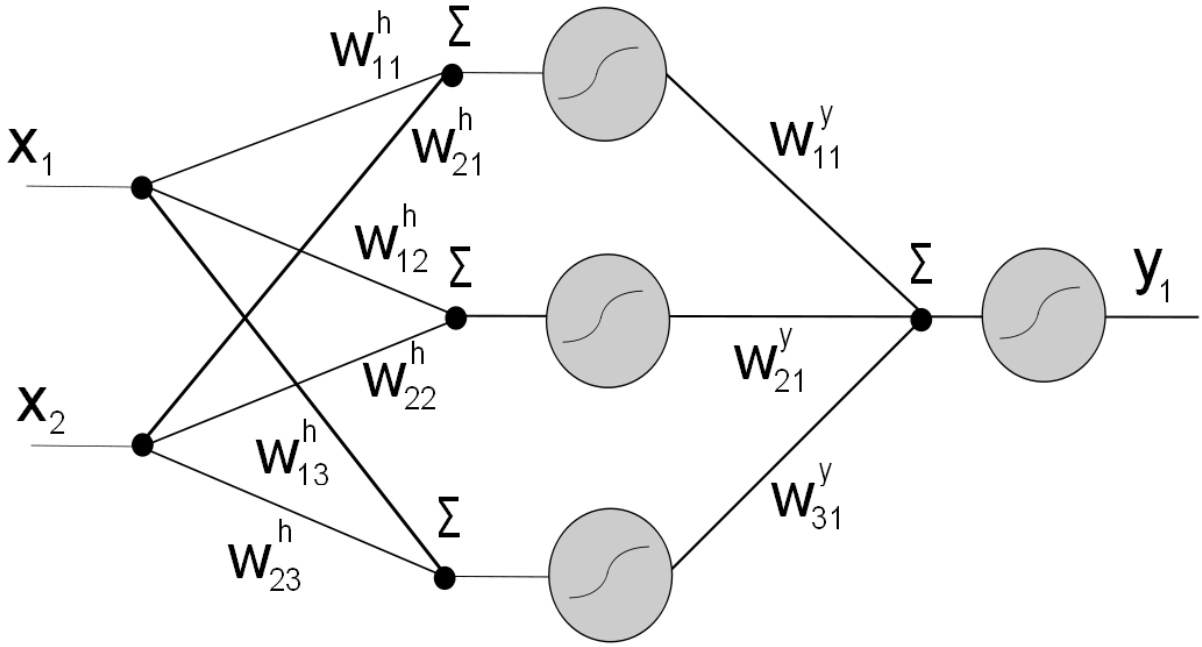

Figure 2.4: A simple multi-layer perceptron.

network is then presented with another data vector and back-propagation repeated for the ensuing error. This continues through the data set. If the network converges successfully it will reach a state of equilibrium where error is reduced to a local minimum. The use of a feed-forward network allows this process to be used without excessive complexity, as it is known that data can only travel one way between nodes [37].

Advantages of MLPs are their ability to "learn" non-linear relationships in multiple input systems, without an understanding of the principles of that system. This is by nature a black-box solution, which is also its chief disadvantage, as it provides no explanation for the outputs and thus makes it difficult to justify them. In addition, as the back-propagation algorithm does not converge to a global minimum, it does not necessarily provide an optimal solution.

2.6.2 Self-Organising Map

A self-organising map (SOM) [38], also known as a Kohonen Map, is a type of ANN used for dimensionality reduction, visualisation and clustering. Often, high-dimensional data is reduced to a 2-dimensional representation. This is made possible by arranging the nodes in a grid arrangement, called a U-matrix. The properties of SOMs allow the relative positions of data in high dimensional space to be mapped to the lower dimensional grid.

The SOM is a collection of interconnected weighted nodes. The training algorithm, a process of vector quantisation, involves comparing the current input vector with the weight vector of each node, using the Euclidean distance as a measure. The best matching unit (BMU) is the node which is the smallest distance from the input vector. The BMU and surrounding nodes' weight vectors are then updated to bring them closer to the input vector. The process is then iterated through for each training input. This process will often result in regions of activity [39], outputting a result which represents groups of data points as vectors around a centre. In this way it is somewhat similar to clustering algorithms like K-means.

An example of the output from the SOM Toolbox for Matlab [40] is shown in Figure 2.5. This shows a SOM being trained to recognise different types of flowers, based on Sepal and Petal length and width. The U-matrix visualises the distribution of values across the four parameters in two-dimensions. The other plots ("PetalL", "PetalW", "SepalL" and "SepalW") are the component planes, and show the kind of values present for each parameter.

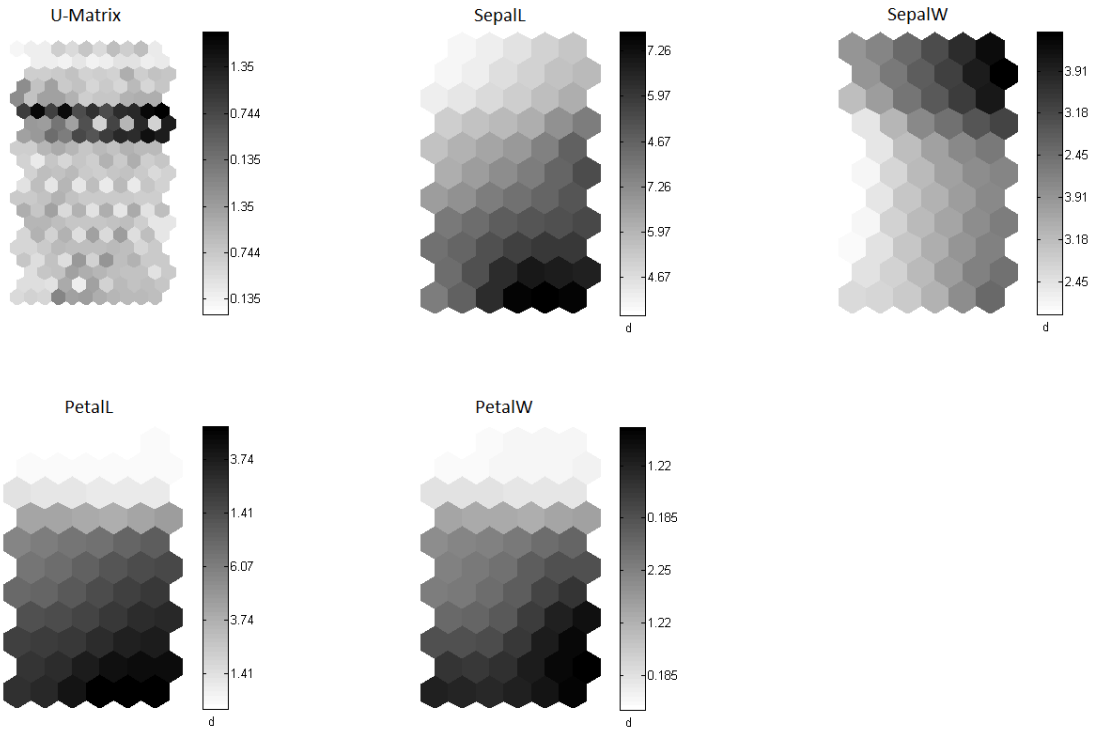

Figure 2.5: An example output from SOM Toolbox, showing U-matrix and component plots.

Figure 2.6 shows the U-matrix after the identified clusters have been visualised. Each cluster represents a kind of flower.

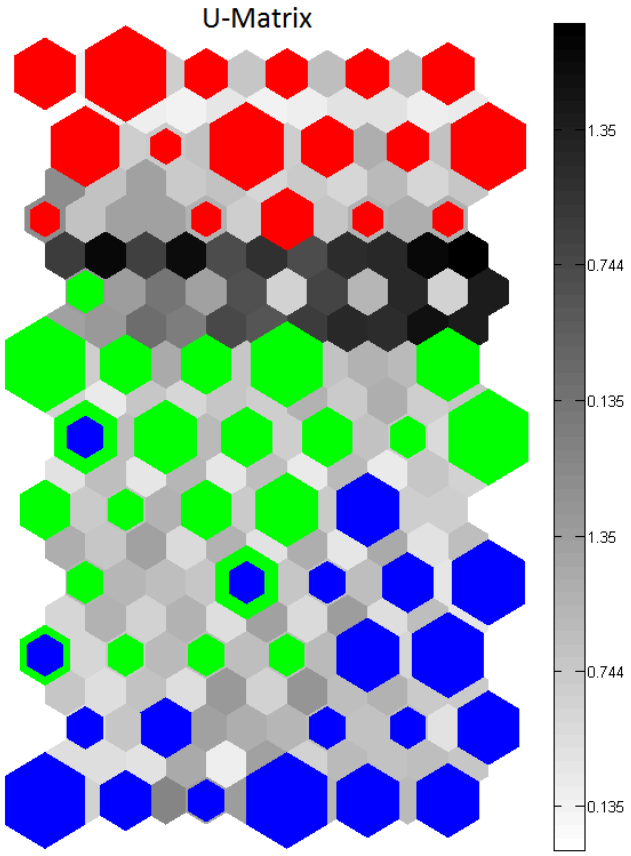

Figure 2.6: The U-matrix showing the identified clusters by colour.

SOMs may be applied to condition monitoring as a way to cluster data. In comparison to K-means and some other clustering methods, SOMs have the advantage of not requiring a priori knowledge.

2.7 Markov Chains

A Markov chain is a probabilistic mathematical model of a system. The system is represented as a process which transisions between a number of states. The probability of transitioning to any state is determined by only the previous state. The Markov process is memoryless, and therefore does not consider any states other than that immediately before. This is called the Markov property (Equation 2.3) [41].

$$P(S_t|S_{t-1}, S_{t-2}...S_0) = P(S_t|S_{t-1}) \tag{2.3}$$

The states of a simple Markov chain are shown in Figure 2.7, with the states given the designations S_1, S_2 and S_3. The state transition probabilities are represented by the nine "a" variables. For a Markov chain with S states, these probabilities are stored in a SxS state transition matrix.

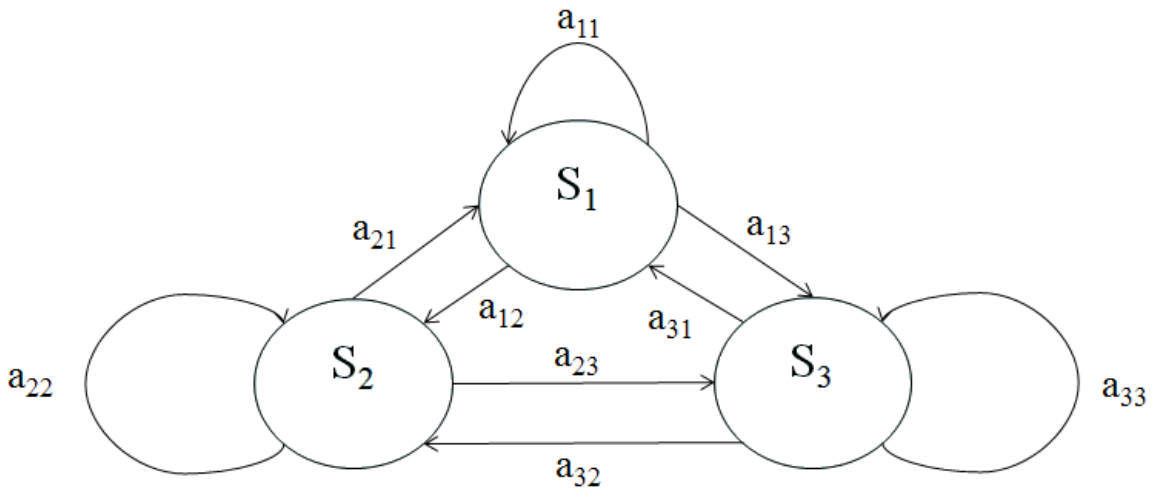

Figure 2.7: State Transition Diagram of a three state Markov chain

Markov chains can be used to model complex dynamic systems [42]. In addition to diagnosis, Markov chains can also be used to model degredation [43] and predict future states in order to provide prognosis [44].

2.8 Hidden Markov Models

Hidden Markov Models [45] are a type of probabilistic model and are an extension of Markov chains. The system is again modelled as a Markov process which transitions between a number of states. However, while the current state of a Markov chain is observable, the HMM's states are hidden. HMMs introduce the concept of an observation, which is a directly observable value. These observations are considered to be related to, but distinct from the underlying state of the system.

An example of the distinction between a state and an observation is possible by looking at the operation of a gas turbine. A simple model of a gas turbine could have the states "run-up", "steady-state" and "run-down". However, these states can not be directly observed but rather can be inferred from the values of the various monitored parameters (pressures, temperatures, rotational speed etc.) of the turbine, which together form an observation.

HMMs are based on the principle that given a set of observations, it is possible to derive the most likely set of underlying hidden states and the associated probability of transitioning between those states. In this way, a HMM may model the internal behaviour of a system based on only that system's inputs and outputs. As with Markov chains they rely on the Markov assumption that the process is memoryless. In hidden Markov models, the probability of transitioning to any state is a function of the previous state probabilities and the current observation.

2.8.1 Defining a HMM

As with markov chains, the state probabilities are stored in a state transition matrix, usually called A, where the a_{ij} element represents the probability of transitioning from S_i to S_j. A HMM also contains a vector with the probabilities of starting ($t = 0$) in a particular state, represented by the vector π, of length S.

Variations exist on how a HMM represents the probabilities of observations. Two types of HMM, with discrete scalar and continuous vector observations respectively, are considered in this thesis. The different approaches to calculating the observation probability is represented in Figure 2.8. Both types are discrete in both time and state. They are explained in greater detail in the following subsections.

2.8.1 a Discrete HMMs

In a discrete HMM (DHMM), the probability of a discrete observation O while in state j, represented by $b_j(O)$, is a scalar value between zero and one. When summed, the probabilities of all observations must be one for each state. If there are O possible discrete observations, B is a SxO observation matrix. A DHMM represented by λ, can now be defined as a tuple:

$$\lambda = (A, B, \pi) \tag{2.4}$$

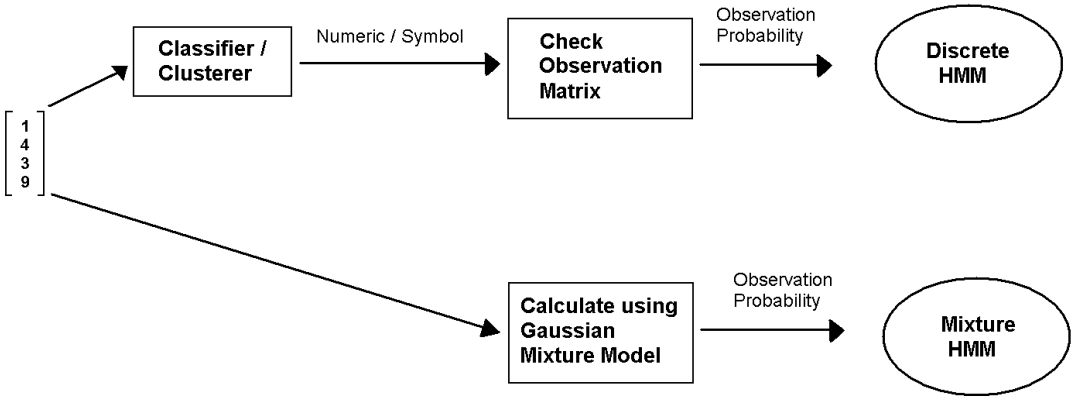

Figure 2.8: A block diagram showing the difference in processes between a discrete and a mixture HMM when calculating observation probabilities

If the original data source contains continuous values, in order to use a discrete HMM it would be necessary to cluster or classify the values in some way to derive discrete observations.

2.8.1 b Continuous HMMs

For HMMs with continuous inputs, O does not represent a discrete value as is the case with DHMMs, but rather a continuous real value. In order to represent the probability of all possible values it is necessary to use a continuous probability density function (PDF), rather than a observation matrix. This is accomplished by combining many Gaussian distributions in a mixture model. This thesis will refer to this type of model as a mixture HMM (MHMM). Equation 2.5 below shows a one-dimensional Gaussian function, where x is the input, μ is the mean and σ^2 is the variance.

$$g(O;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(O-\mu)^2}{2\sigma^2}} \tag{2.5}$$

These distributions are combined using a mixture weight to form an approximate distribution. Each state has its own Gaussian mixture model (GMM). Therefore, if $b_j(O)$ once again means the probability of the observation O while in state j, M is the number of mixtures, and c_{jm} is the mixture weight for the mth mixture of state j, the probability is given by:

$$b_j(O) = \sum_{m=1}^{M} c_{jm} g(O; \mu_{jm}, \sigma_{jm}^2) \tag{2.6}$$

By manipulating the hyper-parameters of the weight, mean and variance of the distributions, a continuous probability density approximation is formed. The observation O, rather than being discretised through clustering or a classifier, is a direct input and a continuous-valued data vector. By determining the probability of an input value directly, without clustering the inputs (although clustering may still be used in the implementation of the algorithm during training to calculate the means for the Gaussian distributions) it avoids the potential loss of sensitivity that discretising the data may introduce. This also serves to mitigate the added complexity of the MHMM compared to its discrete counterpart.

Given D as the dimensionality of O (the input vector), N as the number of states and M as the number of mixtures, the new components can be defined in the following form:

C NxM matrix containing the mixture weights for each state

M NxMxD matrix containing the mean values for each state's mixtures

Σ NxMxDxD matrix containing the covariance matrices (DxD) for the mixtures

A MHMM can now be defined as a 5-tuple containing the transition matrix, mixture weights, means, covariance matrices and prior, shown in equation 2.7.

$$\lambda = (A, C, M, \Sigma, \pi) \tag{2.7}$$

2.8.2 Training and Testing

One of the potential outputs of an HMM, and the primary output utilised in this thesis, is the maximum probability of a particular sequence of observations given the model parameters. As the probability is a function of both the state transition probability and observation probability this is not straightforward to calculate. This complicates train-

ing and testing of the HMM.

2.8.2 a Forward-Backward Procedure

Comparing the probabilities of observation sequences is a potentially computationally expensive task if calculated directly [45]. All possible state transitions must be considered and the probability of the observation sequence must be considered for each set of states, resulting in the number of calculations increasing exponentially with the number of observations. If there are N states, and T observations, then the number of calculations is roughly $2TN^T$, the vast majority of them being multiplications.

Due to the computational complexity of calculating the probability of all but the shortest observation sequences, the direct calculation is not used. The forward-backward procedure [46] [47] incorporates induction to reduce the required number of calculations. This requires the introduction of two terms: the forward variable α, and the backward variable β.

The forward term α is the joint probability of the observations so far and the current state, given the model λ (defined in equations 2.4 and 2.7, depending on the type of HMM).

$$\alpha_t(i) = P(O_1\ O_2 ... O_t, q_t = S_i|\lambda) \tag{2.8}$$

The first (t=1) values are the only ones computed directly. Induction is then used to calculate each subsequent time step, keeping the computational requirements proportional to t.

Initialisation:

$$\alpha_1(i) = \pi_i b_i(O_1), \quad 1 \leq i \leq N \tag{2.9}$$

Induction:

$$\alpha_{t+1}(j) = [\sum_{i=1}^{N} \alpha_t(i) a_{ij}] b_j(O_{t+1}), \quad 1 \leq t \leq T-1, \quad 1 \leq i \leq N \tag{2.10}$$

Termination:

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{2.11}$$

The forward term is all that is needed to calculate the probability of observations se-

quences, and so is used when testing the data with a HMM. The number of calculations to calculate the likelihood of an observation sequence is reduced to N^2T [45], making larger HMMs and observation sequences more plausible. For reference, if we take the values that were used for the models used in the prototype presented in chapter 6 of this thesis (N=10 and T=20), the number of calculations is reduced from 4 x 10^{21} to 2000.

For training and other calculations, the backwards term is required. This is defined as the probability of a set of future observations, given the current state and the model λ:

$$\beta_t(i) = P(O_{t+1}\, O_{t+2}...O_T|q_t = S_i, \lambda) \tag{2.12}$$

Again, this can be calculated through induction. However, this is done backwards from T-1:

Initialisation:

$$\beta_T(i) = 1, \quad 1 \leq i \leq N \tag{2.13}$$

Induction:

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j), \quad t = T-1, T-2, ...1, \quad 1 \leq i \leq N \tag{2.14}$$

2.8.2 b Training the Model

The most complicated process is training which requires the HMM to "learn" the behaviour it is presented with during the training period. This requires some form of algorithm to adjust the HMM parameters so as to maximise the output probability value for behaviour similar to that seen during training. There is no analytic way of determining the optimal values for the HMM parameters. Techniques that can be used to estimate models parameters include the expectation maximisation (EM) algorithm [48] [49] or gradient descent techniques [50] [51].

These algorithms are not guaranteed to produce an optimal result, as they only provide a local maximum, but they are all accepted practical solutions [45]. The Baum-Welsh algorithm was chosen for this thesis [52]. This was chosen as it is a proven technique used by Rabiner in his seminal paper [45]. The Baum-Welsh algorithm is itself a generalised form of EM algorithm [53].

A new term is introduced. This is the probability of transitioning from state i to state j, given the observation and model:

$$\xi_t(i,j) = P(q_t = S_i, q_{t+1} = S_j | O, \lambda) \tag{2.15}$$

$$\xi_t(i,j) = \frac{P(q_t = S_i, q_{t+1} = S_j, O|\lambda)}{P(O|\lambda)} \tag{2.16}$$

The forward (α) and backward (β) terms are again utilised. α is used to work forward to S_i, and β to work backward to S_j. This allows us to rewrite 2.16 as:

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)} \tag{2.17}$$

A further new term, the likelihood of being in state S_i at time t given the model and observation sequence, can be derived by summing ξ over j:

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j) \tag{2.18}$$

Using ξ and γ it is possible to derive re-estimation formulas for the model parameters by calculating the expected values.

$\bar{\pi}_i$ can be considered the expected frequency of being in state S_i at time $t = 1$:

$$\bar{\pi}_i = \gamma_1(i) \tag{2.19}$$

$\bar{a}_{ij}$ can be considered the expected frequency of transitions from state S_i to S_j, relative to the number of transitions from S_i:

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{2.20}$$

$\bar{b}_j(k)$ can be considered the expected frequency of observation k while in state S_i, relative to the number of times in state S_i:

$$\bar{b}_j(k) = \frac{\sum_{t=1,\text{ s.t. O=k}}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} \tag{2.21}$$

While the terms above are sufficient for re-estimating the parameters of a DHMM, it does not consider the additional parameters for a MHMM. However, the GMM parameters can be derived via the same process of expectation maximization as the HMM parameters [54]. The additional parameters are directly related to $\bar{b}_j(k)$ through equation 2.6, and can be derived. As we must consider the contributions of each Gaussian component to the PDF, a modified version of γ must be used:

$$\gamma_t(j,k) = \left[\frac{\alpha_t(j)\beta_t(j)}{\sum_{n=1}^{N} \alpha_t(n)\beta_t(n)} \right] \left[\frac{c_{jk} g(O_t; \mu_{jk}, \sigma_{jk}^2)}{\sum_{m=1}^{M} c_{jm} g(O_t; \mu_{jm}, \sigma_{jm}^2)} \right] \tag{2.22}$$

Summing $\gamma_t(j,k)$ over all the mixtures would calculate the probability of being in state S_j at time t across the entire PDF, resulting in an equivalent result as the previous $\gamma_t(j)$ term:

$$\gamma_t(j) = \sum_{k=1}^{M} \gamma_t(j,k) \tag{2.23}$$

The re-estimation formulas are:

$$\bar{c}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)}{\sum_{t=1}^{T} \sum_{m=1}^{M} \gamma_t(j,m)} \tag{2.24}$$

$$\bar{\mu}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k).O_t}{\sum_{t=1}^{T} \gamma_t(j,k)} \tag{2.25}$$

$$\bar{\sigma}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k).(O_t - \mu_{jk})(O_t - \mu_{jk})'}{\sum_{t=1}^{T} \gamma_t(j,k)} \tag{2.26}$$

The re-estimation formulas for $\bar{\pi}_i$ and $\bar{a}_{ij}$ are the same for both types of HMM considered here. Using these equations we can calculate the re-estimated model $\bar{\lambda}$. The re-estimated model is then used as λ in the next iteration, which produces a further refined set of parameters. Once $\bar{\lambda} = \lambda$ the model has reached a (local) maximum and training may stop.

2.8.2 c Calculating state sequences

While not necessary for calculating the probability of a particular observation sequence, it can often be useful to determine the most likely sequence of underlying states, given a set of observations. This is possible through the Viterbi algorithm [55]. This was not implemented in the prototype presented in this thesis, but would be a useful addition in future iterations.

2.8.3 Interpreting the model output

When considering the output probabilities it is necessary to use a logarithmic scale. This is because, being a probability, the output of the HMM is continually multiplied by values less than 1. Figure 2.9 shows that this makes it difficult to interpret the output after only 50 observations of a 100 observation sequence. By using a log scale the curve becomes a roughly straight line for normal behaviour.

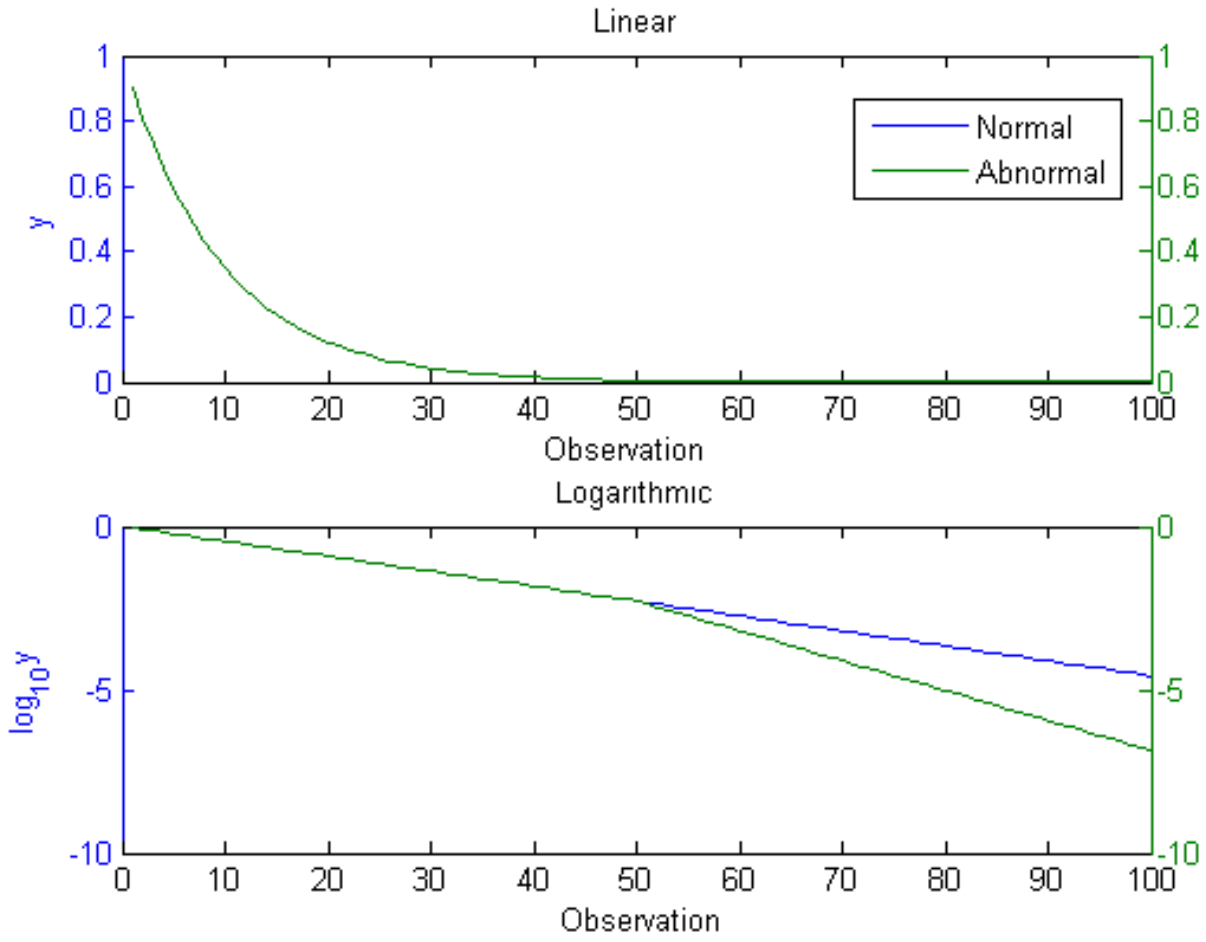

Figure 2.9: Linear vs. Log Scales for Normal and Abnormal Sequences

The log scale also makes it much easier to pick out anomalies (which present themselves as deviations from the trajectory of normal behaviour) and allows easier visualisation of the results. In general, when considering the output of the HMM, the term "likelihood" is used in literature [45] rather than "probability", and as the log scale is used, this is then termed the log-likelihood (LL).

A useful way to visualize the results of a HMM is to plot several sequences simultaneously, as in the work by Fox *et al.* [56]. This allows a "cone" of normality to be built up, allowing sequences that diverge from normal behaviour to be easily identified. However, it can also be desirable to plot the LL over a long period of time. In this case, only the final LL value is presented to the user. A comparison of the two visualisations is provided in Figure 2.10.

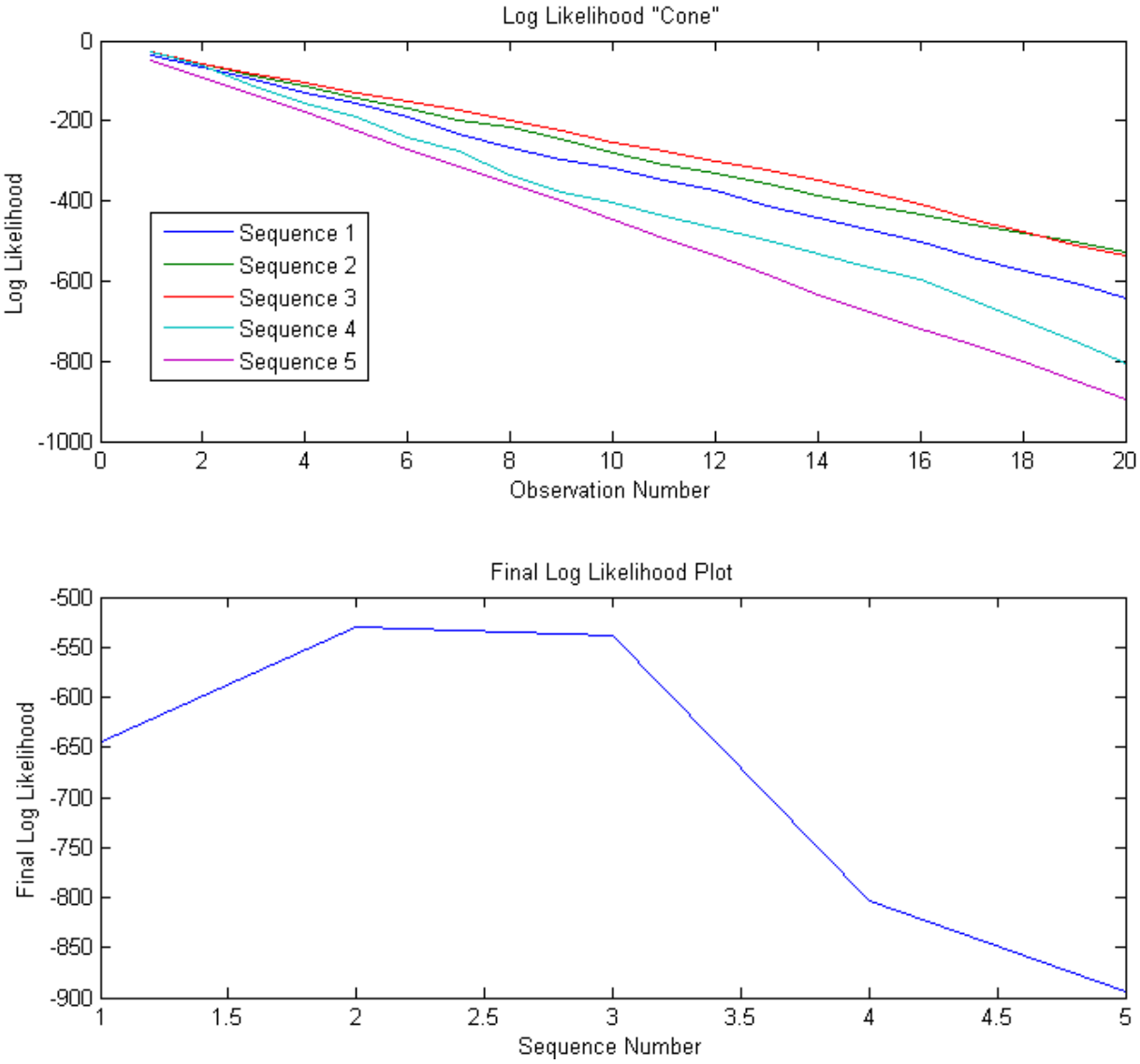

Figure 2.10: Comparison of Cone and Final Log Likelihood Plots

2.9 Linear Regression

Linear regression is the process of fitting a line of best fit to a set of data [57]. In this way, it may be possible to show a trend without noise and large changes due to transients. The least squares algorithm provides this capability while maintaining relative simplicity and transparency.

The aim is to derive values for a and b, where they are the standard coefficients for a straight line (equation 2.27) that minimises the error between actual and fitted values

of y.

$$y = a + bx \tag{2.27}$$

$$E^2 = \sum_{i=1}^{n} (y_i - (a + bx_i))^2 \tag{2.28}$$

The sum of the squared errors is used (equation 2.28) because this ensures all errors are greater than zero and allows the sum to be treated as a continuous differentiable quantity. This does mean that outliers have a stronger effect than otherwise, and this must be considered when deciding if this algorithm is appropriate for any particular application. The sum of the error is minimised when its rate of change is zero. This results in equations 2.29 and 2.30.

$$\frac{d(E^2)}{da} = -2 \sum_{i=1}^{n} [y_i - (a + bx_i)] = 0 \tag{2.29}$$

$$\frac{d(E^2)}{db} = -2 \sum_{i=1}^{n} [(y_i - (a + bx_i))x_i] = 0 \tag{2.30}$$

It is possible to derive the following equations for a and b [58]:

$$a = \frac{\bar{y} \sum_{i=1}^{n} {x_i}^2 - \bar{x} \sum_{i=1}^{n} x_i y_i}{\sum_{i=1}^{n} x_i^2 - n\bar{x}^2} \tag{2.31}$$

$$b = \frac{\sum_{i=1}^{n} x_i y_i - n\bar{x}\bar{y}}{\sum_{i=1}^{n} x_i^2 - n\bar{x}^2} \tag{2.32}$$

Using equations 2.31 and 2.32 we can now calculate the a and b values for a line that fits the data while minimising the error. An example is shown in Figure 2.11. The Figure shows data sampled from the line $y = 50 + 3x$, with Gaussian noise added.

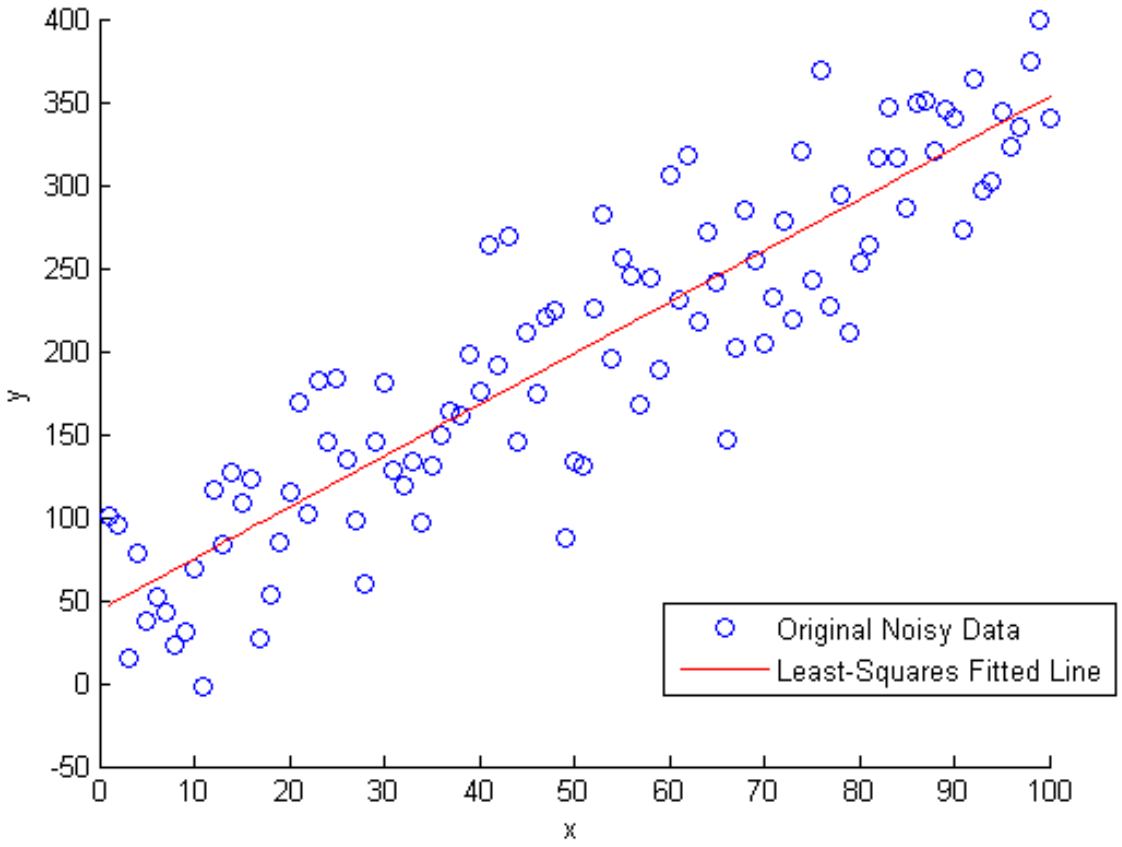

Figure 2.11: Example of least-squares line fitting applied to generated data.

The least squares algorithm calculated a = 44.0953 and b = 3.0883. While the y-intercept (a) is a little low, this is a reasonable approximation. As the gradient (b) is of greater importance when determining the presence of degradation, rather than the absolute value, this is an acceptable approach for the application presented in this thesis.

2.10 Knowledge-based Systems

Two of most prominent forms of knowledge-based problem solving are rule-based reasoning, implemented as part of rule-based expert systems, and case-based reasoning (CBR) systems [59] [60]. Rule-based expert systems (RBESs) represent the knowledge of domain experts as a set of rules. This can take the form of a series of IF... THEN... structures in a rule-base [61]. CBR systems are instead based on the system recognising similarities between observed inputs and a set of examples, or "cases". Where the examples are known fault cases the system is able to determine which case the current input most resembles, and provide the appropriate fault diagnosis [62]. Some systems, such as NEST, utilise a hybrid approach with aspects of both types of system [59].

In this thesis, RBESs were chosen to implement captured knowledge. This was due to an insufficient number of example cases being available, making a case-based reasoning system impractical. KBSs take advantage of the domain knowledge of a set of experts, which were available to provide the required knowledge. The following section describes the knowledge capture methodology, and the principles of RBESs.

2.10.1 Knowledge Engineering

The process of developing expert systems was originally thought to be the act of transferring the knowledge of an expect directly into a software implementation [63]. However, this approach proved difficult to scale up to large industrial problems [64]. Similarities have been drawn between these early expert systems and early computer programs, before the "art" of developing such systems was formalised as software engineering. Similarly, the discipline of knowledge engineering has emerged, and is now generally considered to be a process of creating a computer model with similar problem solving capabilities to a domain expert [64]. The process of developing this model has been formalised in a number of structured methodologies [65]. Examples of such methodologies and tool kits to support them include CommonKADS [63], Model-based and Incremental Knowledge Engineering (MIKE) [66], and Protégé [67].

CommonKADS was selected because it has already been successfully applied to capture knowledge of experts in the power domain [68]. CommonKADS is the third version of the Knowledge Acquisition and Documentation Structuring (KADS) methodology [69]. It is well supported by several workbenches designed to assist a knowledge engineer in what can often be a time-consuming and labour intensive task [70].

The full CommonKADS framework consists of six models:

Organisation Model
Describes the structure of the organisation, and identifies the potential impact of KBS development.

Task Model
Describes the tasks of the KBS within the organisation's structure.

Agent model
Describes the characteristics of agents. Agents, human or not, are the executors of the tasks.

Communication model
Representation of the communication between agents.

Knowledge model
An implementation dependent representation of the knowledge necessary to perform a task.

Design model
The structure of the system. Specifies how the knowledge and communication models are implemented.

The knowledge model and the design model are the only models required to describe the functional aspects of a KBS [64]. This thesis only considers the knowledge model. This is sufficient to represent the knowledge captured. However, a full design model is necessary before the knowledge may be implemented as a deployable system.

The knowledge model breaks the captured knowledge into three types [71]:

Task Knowledge
The hierarchy of tasks and sub-tasks that must be completed in order to complete a goal.

Inference Knowledge
The inferences, using domain knowledge, that are required in order to complete a task.

Domain Knowledge
Knowledge about the field in question. May be explicit or tacit expert knowledge.

In order to capture this knowledge, a four stage process is followed.

2.10.1 a Knowledge Elicitation

The process through which the expert knowledge is captured. CommonKADS proposes five approaches to elicit knowledge from an expert [63]:

Interviews
Either Structured or Unstructured

Shadowing
The knowledge engineer will observe the expert as they work

Laddering
An informal "tree" of associations between the various domain objects is built up

Card Sorting
The expert is presented with a set of cards with domain objects written on them, and asked to categorise them

Repertory Grids
The expert is presented with a set of similar domain objects, and asked to find the objects that do not fit. Through statistical and clustering processes it is possible to derive new groupings of objects.

From these processes the knowledge of the experts is captured. However, it is not yet human or machine readable, and must be represented in an appropriate manner before it can be validated by the experts and ultimately utilised as part of a KBS.

2.10.1 b Knowledge Representation

The knowledge is consolidated into a series of meeting transcripts. Graphical representation is also typical through the Universal Modelling Language (UML) diagrams [72].

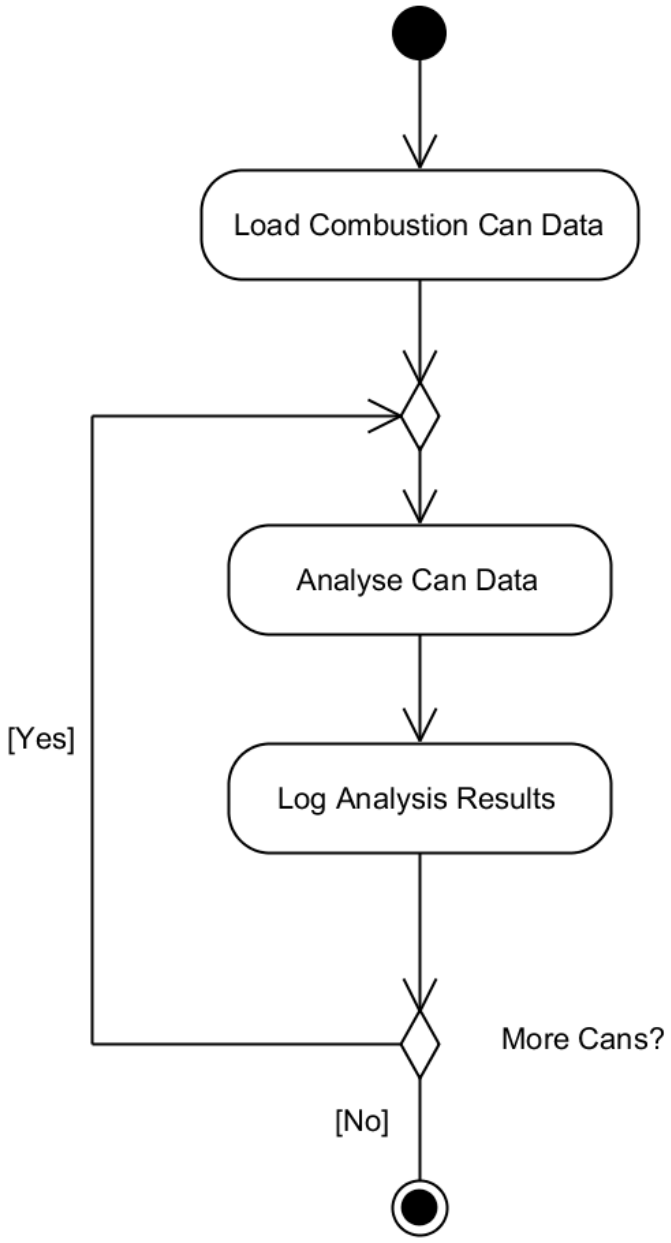

Figure 2.12: An example activity diagram showing a potential combustion can condition monitoring strategy.

An activity diagram can be useful for showing the actions an expert may perform in order to complete a goal. This also allows visualisation of any decisions that may be taken, and the data flow throughout the task. Figure 2.12 shows an example of the actions performed when monitoring the combustion cans of a gas turbine.

A task diagram (Figure 2.13) shows a break-down of the tasks and sub-tasks an ex-

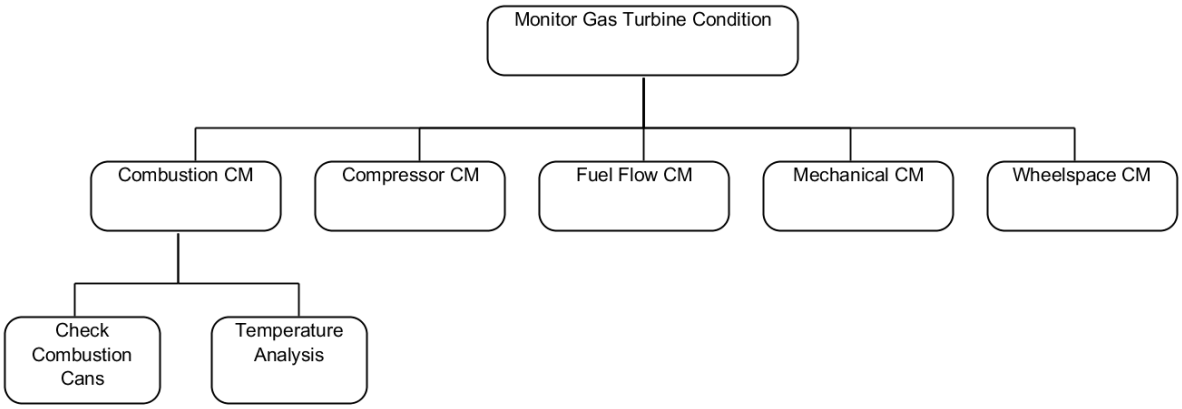

Figure 2.13: An example task diagram showing one possible approach to the condition monitoring of a gas turbine.

pert must complete in order to complete a particular goal, in this case monitoring of the full gas turbine across several sub-systems. This task hierarchy takes on a tree structure, where each "branch" below must be completed before the task is considered complete.

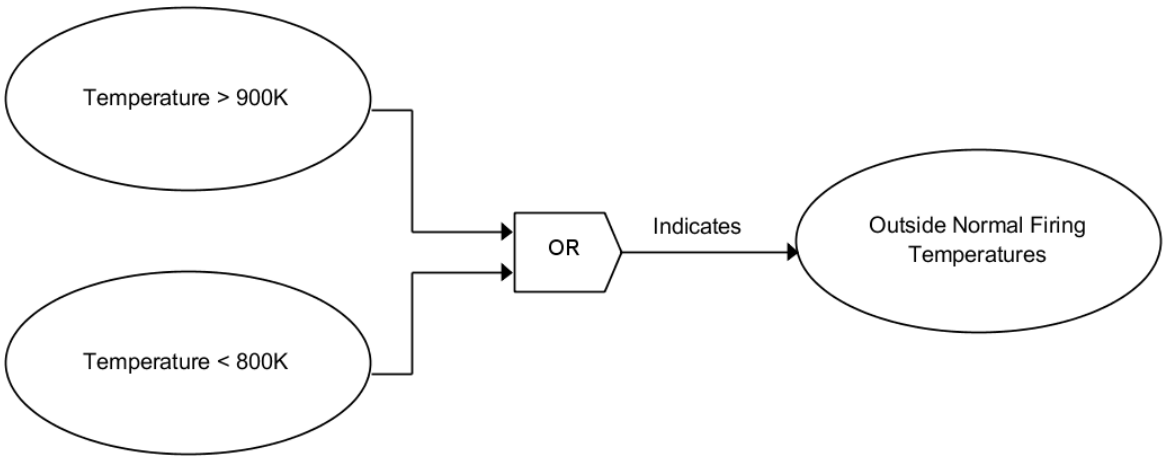

Figure 2.14: A basic semantic diagram representing a check on combustion temperatures.

A semantic diagram shows the low level rules captured during the knowledge elicitation. The rules represented in the semantic diagrams are a graphical representation of those that will ultimately form the rule-base. Figure 2.14 shows an example of a semantic diagram representing a check on the firing temperature of a combustion can.

2.10.1 c Knowledge Validation

A back-and-forth process where inaccuracies or ambiguities in the captured knowledge may be addressed. The experts will provide feedback on the meeting transcripts and diagrams produced by the knowledge engineer. This may result in further elicitation to address gaps in the captured knowledge, or corrections to the knowledge representations. The steps of elicitation, representation and validation are repeated until the captured knowledge is considered complete and accurate.

2.10.1 d Knowledge Utilisation

The knowledge is converted into a set of rules and implemented as a KBS in a programming environment. The CommonKADS methodology is designed so that the representation of the data does not tie the engineer into a specific implementation environment.

2.10.2 Rule-Based Expert Systems

A rule-based expert system is a type of expert system where the domain knowledge is represented as a series of "**if** *premise* **then** *conclusion*" rules. This is known as the knowledge base. The rules are traditionally represented in first-order logic, but systems exist where fuzzy logic [73] is instead used due to its ability to represent uncertainty [74].

In addition to the rule-base, a RBES requires an inference engine. The inference engine uses the information in the knowledge base to draw any valid conclusions. The algorithm used to perform this task may be either forward-chaining, or backward-chaining [37].

Forward-chaining

The algorithm begins when a new piece of knowledge is added to working memory. The knowledge base is then searched for any rules where the new piece of knowledge is a premise. Any new conclusions that are now true, are also added to working memory. This process is repeated until it is not possible to add any additional conclusions. This process can lead to a number of conclusions, not all of which may be relevant.

Backward-chaining

In the case of backward-chaining, the algorithm is designed to "answer" questions

posed to it. The algorithm is iterative in a similar way to forward-chaining. Rather than adding a premise and propagating this through the rules, a conclusion is provided and the algorithm works backwards through the rules to determine if the necessary premises are met for the answer to be true.

RBESs have advantages compared to machine learning techniques in that the logic behind arriving at a conclusion can be provided [75]. This allows the steps in reaching any conclusion by a RBES to be inspected and justified, as opposed to machine learning techniques which are often "black box" approaches. Also, due to the separation of inference engine and knowledge-base, it is possible for the knowledge in the system to be updated and extended. While a machine learning algorithm can be re-trained, this can be time consuming and is often required to be done off-line. The training algorithm may also not be deterministic, making re-training less desirable. Rule-based expert systems may also be possible where large amounts of data are not available, where machine learning algorithms would not be feasible.

The main disadvantage of RBESs is the inability to model anything that cannot be understood by a human expert. Machine learning techniques may be able to detect faults based on patterns in the data that even experts may not be able to link to faults. Essentially, the problem must already be solved before the RBES can then automate the process as a set of rules. This makes expert systems inappropriate for new fields or any other circumstance where insufficient domain knowledge is available [61].

An expert system lacks any deep understanding of the problem domain. This is because many of the rules may be from the experience of the domain expert, and may be heuristic in nature. Thus, no explanation of why a particular conclusion may be reached based on the premises for any given rule is provided. Whereas an expert can reason on a problem from first principles if their "rules-of-thumb" fail to provide a solution, an expert system will simply fail. For the same reason, while an expert system can explain how a certain conclusion was reached through the inferences made at each step to reach the final conclusion, it cannot explain why. This lack of deep understanding is part of the reason why expert systems are difficult to apply to more general "common sense" reasoning applications [61].

2.11 Summary

This chapter has provided an outline of the data mining methodology, and a description of the types of AI algorithms. The remainder of the chapter has reviewed a number of techniques that may be applicable to the condition monitoring of plant assets.

Chapter 3

Multi-Agent Systems

Multi-agent systems (MASs) are an area of AI that involves developing systems as a collection of separate entities known as agents. There are many definitions about what a MAS is, both within the AI field and outside [76]. This chapter will define what constitutes a multi-agent system, and describe the technology utilised in realising the prototype multi-agent system (Chapter 6).

3.1 Definition of an Agent

Before we can define what a MAS is, a definition for an agent is necessary. The seminal work by Wooldridge and Jennings defines an intelligent agent. This can be considered to be an agent that displays the following core properties [77]:

Autonomy	The agent can operate effectively without prompting from external sources such as a user, and has control of its own internal state.
Reactivity	The agent can perceive and react to changes in the surrounding environment, taking actions as appropriate to the circumstances.
Pro-activeness	The agent is goal orientated, and actively seeks out the means to achieve those goals. Wooldridge describes this as "taking the initiative"
Social ability	The agent has the ability to co-operate with other agents to achieve common goals.

This definition is used for all agents referred to in the work of this thesis.

3.2 Definition of a Multi-Agent System

A MAS can be defined as a collection of one or many agents working together towards common goals.

The agents within the multi-agent system may have one of many roles, which may evolve over time. The social interaction between agents means that the relationships between, and the role of individual agents may change dynamically depending on the current goal, the state and number of other agents, or the current conditions [78]. The agents within the system can therefore configure themselves without interaction with the user, to achieve the objectives of the system as a whole. These properties provide the MAS with a high degree of pro-activeness and flexible autonomy.

3.3 Technology

The Foundation for Intelligent Physical Agents (FIPA) is a standards organisation for MASs. It has been accepted as a standards committee for agent technology by the Institute of Electrical and Electronics Engineers (IEEE) Computer Society, and provides a

number of specifications for the development of MASs with the properties and advantages cited in Sections 3.1 and 3.2 [79].

3.3.1 Agent Platform

An agent platform is what allows agents to successfully deploy, operate and communicate with each other. Without an effective means to communicate between agents, the social-ability of the agents is compromised. The platform must also allow for additional agents to be added to the system, in order for the flexibility of the MAS to be maintained. It is therefore fundamental to the development of an operational MAS. Figure 3.1 shows the FIPA Agent Management reference model [80]. This shows the high-level structure of the agent platform.

The platform uses a message transport service and two utility agents in order to allow application agents to join and contribute to the MAS:

Message Transport System (MTS)	The means by which agents registered on the platform may send messages and communicate with each other. It also permits communication between agents across FIPA-compliant platforms.
Agent Management System (AMS)	The "white pages" of the platform. A utility agent that maintains a list of the agents on the platform and provides supervisory control.
Directory Facilitator (DF)	The "yellow pages". A utility agent that maintains a directory of services to which agents can both advertise their own services and look-up agents which provide services it requires.

In addition to these platform-provided services, user-designed application agents are deployed. These provide the desired functionality required of the particular MAS.

3.3.2 Communication

The MTS provides the means to send messages across the platform, allowing communication between agents. However, agents are only able to understand one another if

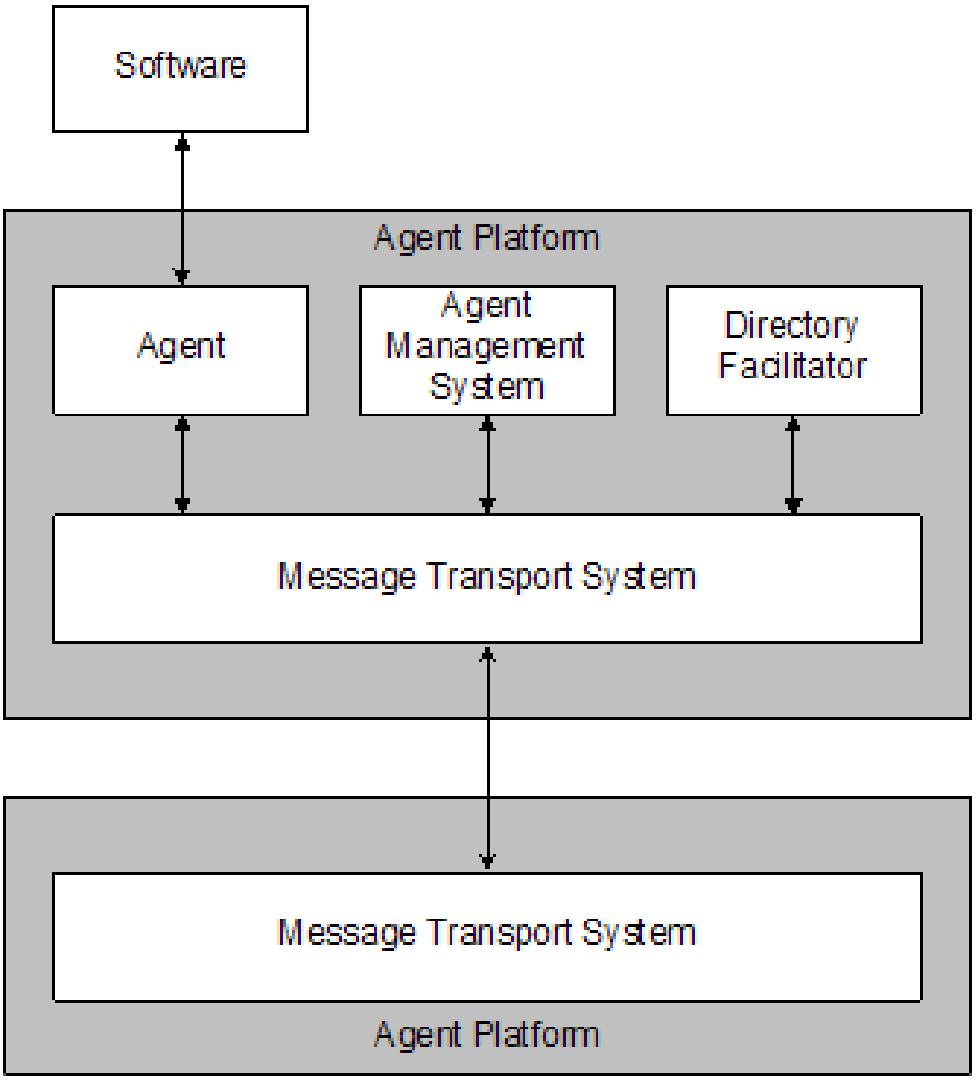

Figure 3.1: The Agent Management Reference Model [80].

they "speak" a common Agent communication language (ACL), content language, and share a common ontology. The ACL provides the structure for the messages, the content language defines the semantics, while the ontology defines the vocabulary.

3.3.2 a FIPA Agent Communication Language

The FIPA Agent Communication Language (FIPA-ACL) specification [81] ensures a common message structure in order to allow agents to understand and negotiate with each other. A message contains a set of parameters, shown in Table 3.1.

The type of communicative act is determined by the performative. This is based pri-

Table 3.1: FIPA ACL Message Parameters with Description

Parameter	**Description**
Performative	Type of communicative act. Possible acts specified in the FIPA-ACL Communicative Act Library Specification [82].
sender	Identity of the message sender.
receiver	Identity of the message receiver.
reply-to	Agent to which replies and further messages within a conversation should be directed to.
content	Content of message.
language	The content language used.
encoding	Encoding of message content.
ontology	The ontology used.
protocol	Interaction protocol used. This defines the structure of the conversation, as defined in FIPA Request Interaction Protocol [83].
conversation-id	ID for this conversation. This allows an agent to carry out several conversations concurrently, with a single or many agents.
reply-with	The expression to be used when responding to this message.
in-reply-to	The previous message to which the current message is a reply.
reply-by	The date a reply must be received by.

marily on speech act theory [84], and includes linguistic terms such as "agree", "refuse" and "request", to name a few. In theory, an agent should be able to respond to all 22 of the defined performatives, but this is rarely the case in practice.

While the structure of the message has been defined, and the MTS is capable of transporting it, the content of the message may not be understood by the receiving agent unless the agents share a common content language. The agents are able to determine the content language of the message using the language field. If agents are unable to understand a received message they can return a message with the performative "Not Understood".

3.3.2 b Content Language

The content language, as the name suggests, defines the structure and format that the content of a message must take in order to allow agents to decipher their meaning. There are several content languages, but at the time of writing the only FIPA-compliant language to go beyond the experimental stage is the FIPA Semantic Language (FIPA-SL) [85].

FIPA-SL content expressions (so called because they form the content of the ACL

message) are made up of well-formed formulas (WFFs), constructed from atomic formulas by applying one of several operators, defined in the specification. The specification splits the operators into three subsets, with SL0 representing the required minimum subset, with SL1 and SL2 adding to the previous subsets' operators to increase the expressiveness and potential power of the language.

Below is an example of the use of the all operator, where the agent "ControlRoom" queries "Substation1" for all assets behaving abnormally. The substation has the knowledge: KB=normalBehaviour(A), anomalousBehaviour(B), normalBehaviour(C), normalBehaviour(D), anomalousBehaviour(E), offline(F).

(query-ref
:sender (agent-identifier :name ControlRoom)
:receiver (set (agent-identifier :name SubStation1))
:content
"((all ?x (anomalousBehaviour ?x)))"
:language fipa-sl
:reply-with query-1)

(inform
:sender (agent-identifier :name SubStation1)
:receiver (set (agent-identifier :name ControlRoom)
:content
"((= (all ?x (anomalousBehaviour ?x)) (set(B)(E))))"
:language fipa-sl
:in-reply-to query-1)

The expression "all ?x (anomalousBehaviour ?x)" can be read as "all x for which anomalousBehaviour is true of x". As "B" and "E" satisfy the proposition, the substation informs the control room with a set containing references to those objects.

3.3.2 c Ontology

An ontology is the vocabulary used in any conversation. The ontology will typically vary between applications. While FIPA specifications clearly specify standards for message structure and content, this is not the case with the ontology. If a suitable ontology

does not already exist, which can often be the case when developing an agent system in a new domain, it is necessary to create a new ontology which can be time consuming.

Tools exist that speed the creation of ontologies [86]. As well as being a knowledge-base framework, Protégé (previously mentioned in section 2.10.1) is also an ontology editor. Protégé [87] provides a graphical user interface (GUI), shown in Figure 3.2, to construct the ontological concepts.

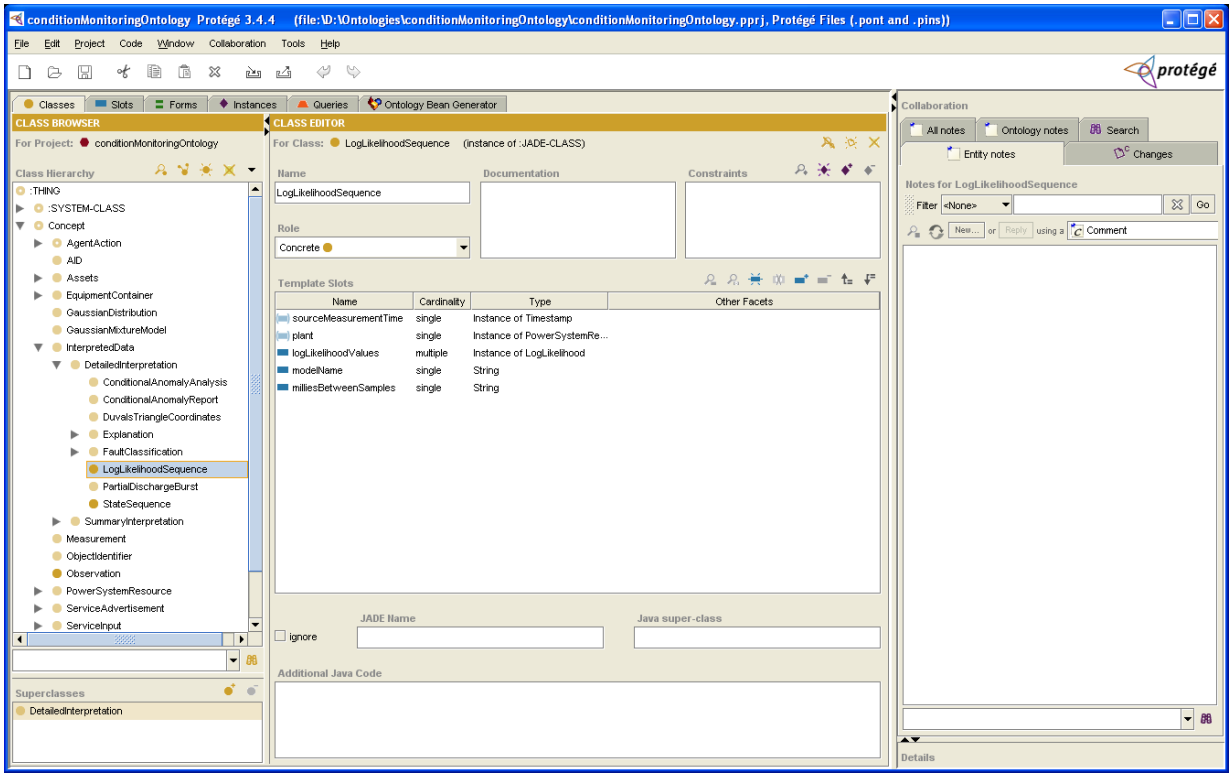

Figure 3.2: Protégé Ontology Editor

3.4 Agent-Orientated Programming

The following section describes the agent-orientated programming (AOP) paradigm and the implementation chosen to allow development of a practical MAS.

3.4.1 The Agent-Orientated Programming Paradigm

AOP can be seen as an extension of object-orientated programming (OOP) [88]. Table 3.2 shows how the two paradigms compare to one another.

Table 3.2: OOP versus AOP [76]

	OOP	AOP
Basic unit	object	agent
Parameters defining state of basic unit	unconstrained	beliefs, commitments, capabilities, choices ....
Process of computation	message passing and response methods	message passing and response methods
Types of message	unconstrained	inform, request, offer, promise, decline ....
Constraints on methods	none	honesty, consistency ....

There are two primary differences between agents and objects. While an object has no constraints on how it represents its state, an agent follows a set architecture to represent a set of mental components. An example would be the belief, desire, intention (BDI) architecture, which defines its three mental attributes using modal logic [89]. Secondly, while the types of messages to objects are unconstrained, the messages passed agents between agents must follow an ACL.

3.4.2 Java Agent Development Framework

The Java Agent DEvelopment framework (JADE) is a Java-based implementation of the FIPA specifications [90]. It extends Java to allow the use of an AOP paradigm. In order to do this it implements two new classes - "Agent" and "Behaviour". By extending these classes the user is able to create new agents and their behaviours (analogous to tasks). Three primary behaviour types are provided: one-shot, cyclic and generic. The use of Java as an implementation language allows the development of platform independent agents for any system that runs a Java Virtual Machine (JVM).

JADE provides a Remote Monitoring Agent (RMA) to allow the monitoring of agents and the messages being passed. It also provides a GUI, shown in Figure 3.3, to allow easy viewing of this information. In addition, it provides several system control functions through this GUI, such as starting or terminating agents, or sending messages to agents.

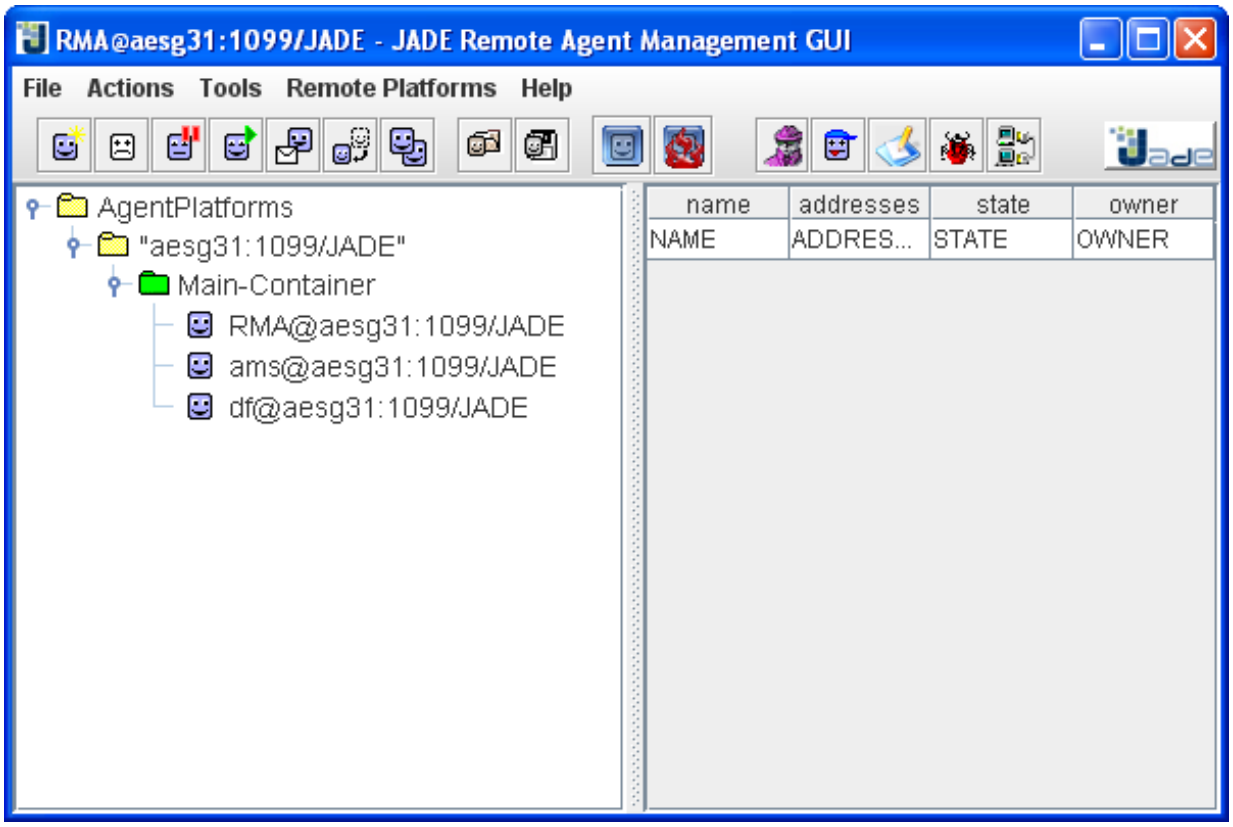

Figure 3.3: The JADE GUI

3.5 Multi-Agent Systems for Real-World Applications

The following section explores the deployment of MASs for industrial applications, with a focus on the power engineering field. The examples provided show that multi-agent systems are sufficiently mature and reliable to make them suitable for monitoring high value assets, where a failure could lead to a large loss in revenue.

The IEEE Power and Energy Society, through the Multi-Agent Systems Working group, has produced a two-part white paper that attempts to define a framework for multi-agent development for power applications [91] [92]. The white paper shows that MASs are applicable to the power engineering domain [91], with standards and tools available [92].

A MAS can provide flexibility and the capability to allow multiple models to run concurrently. Due to the use of standardised inter-agent communication protocols and a platform-independent agent environment the system can also be deployed regardless of platform or network topology, providing a level of future-proofing.

3.5.1 Multi-Agent Systems for Complex Engineering Systems

Work by Jennings et al. evaluates MASs as an approach to developing complex engineering systems [78]. He provides two examples of successful multi-agent systems. The first is the ARCHON architecture [93]. The primary focus of the ARCHON architecture was to allow the introduction of distributed AI to large and complex industrial applications. It was designed to allow the co-ordination of several intelligent systems, with a particular focus on incorporating existing expert systems.

ARCHON allows existing systems to co-operate in a loosely coupled agent system, rather than requiring re-engineering of existing software to permit close integration. Each ARCHON agent consists of the existing intelligent system, or "domain layer", and the ARCHON layer, implemented as a knowledge-based system, and made up of several modules [93]. The modules are:

High-Level Communication
: Facilitates communication between agents. Includes intelligent addressing capability.

Agent Information Management
: Stores the agent's knowledge. This includes models of other agents and itself. They represent the skills, interests and goals of each agent. This allows agents to anticipate requests for co-operation from other agents.

Planning and Co-ordination
: This schedules agent behaviours and deals with any conflicts that may occur.

ARCHON has been successfully utilised by Iberdrola for its transmission control decision support system [94]. This took the form of a MAS consisting of seven agents running across five machines, and performance of the MAS was found to be superior to any one of the previously separate components operating in isolation. The MAS was also found to be more robust and in some cases faster, while providing an integrated interface that was previously not available.

Jenning's second example is of a MAS developed by Daimler (formerly Daimler-Chrysler) [95]. This was used to control the routing of pallets through an assembly line. It is suggested that a centralised approach is impractical for such a large and complex system, providing a performance bottleneck and a single-point of failure [78]. The system instead utilised agents to represent the machines and pallets using an auction

system to allocate machine time. In addition, each switch on the production line (a point at which a pallet can be redirected) was also represented by an agent, allowing pallets to negotiate with a switch in order to be redirected to avoid bottlenecks. The system was successfully deployed and was proven to avoid deadlock and provide near-optimal performance.

3.5.2 Multi-Agent Systems for Power Plant Control

Several systems have been proposed by Lee et. al for the monitoring and control of power plant. An example proposes a MAS utilising "Feedback Control Agents", each using genetic algorithm-based fuzzy control, along with other autonomous agents to provide control to a fossil-fuel power plant [96]. Lee outlines several intelligent techniques, and proposes a Multi-Agent System Intelligent Control System employing them [97]. The agents' robust, reactive and proactive properties as cited as well suited to dynamic and unreliable situations, such as those found in power plant control. Further work adapts the proposed MAS architecture to develop an Intelligent Reference Governor [98] and applies it to a 600MW oil-fired unit, showing the adaptability of such systems and architectures.

3.5.3 COndition Monitoring Multi-Agent System

COndition Monitoring Multi-Agent System (COMMAS) originally started as a GT monitoring system [99], specifically for 3-spool aero engines. The principle was to use multiple agents, each monitoring a different phase of GT operation (shut-down, start-up, part-load, full-load, run-down etc.). The agents themselves would not change, but rather the knowledge base they use would change depending on the phase they were assigned to monitor. The Zeus agent tool-kit was used to implement the system, and the Knowledge Query and Manipulation Language (KQML) agent communication language was used.

The eventual application of COMMAS was in the fault diagnosis of transformers using partial discharge (PD) data [100]. COMMAS was subsequently transferred to the JADE platform, and utilised FIPA-ACL and fully implemented the FIPA specifications [101]. This system shows the extensibility of the multi-agent architecture, having been extended multiple times. Notable extensions include an anomaly detection capability through the addition of an HMM agent [5], and a knowledge-based system to comple-

ment the existing machine learning algorithms [102]. The system has also been shown to operate successfully using several evidence combination techniques [103]. These, and other extensions, have been implemented as additional agents with limited or no modification to the existing agents. A version of the architecture, featuring an anomaly detection agent at the data monitoring stage, is shown in Figure 3.4.

The agents of COMMAS were integrated with several other agents as part of a larger and more comprehensive condition monitoring system [104]. The use of multiple algorithms, each deployed as agents, and the use of evidence combination directly influences the design of the prototype system presented in this thesis.

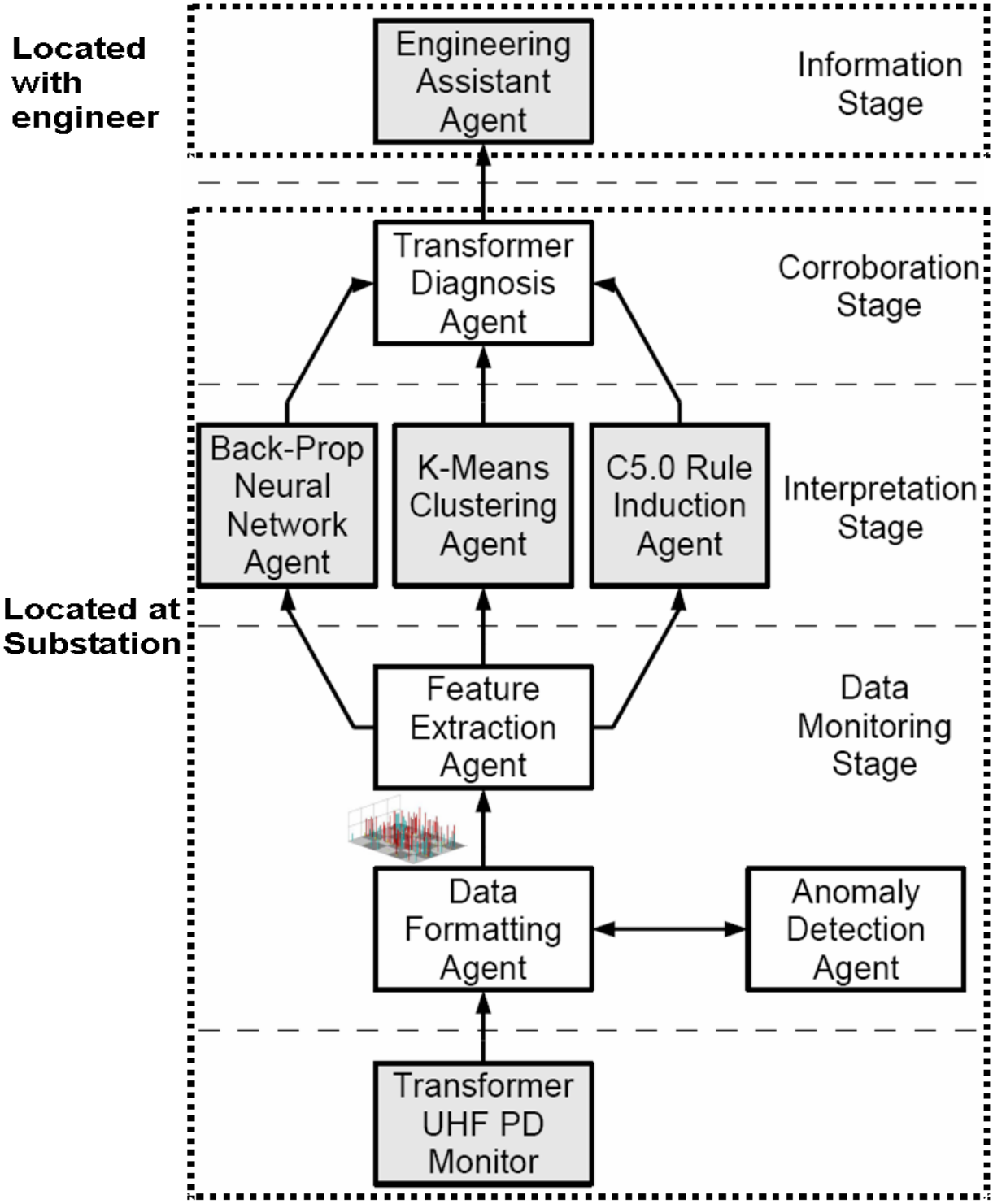

Figure 3.4: The Architecture of COMMAS [5].

3.5.4 Protection Engineering Diagnostic Agents

Protection Engineering Diagnostic Agents (PEDA) is a MAS designed to automate analysis of supervisory control and data acquisition (SCADA) and digital fault recorder (DFR) data [105]. It consists of three legacy intelligent systems, each encapsulated by an agent container. PEDA integrates these tools and automates analysis through the addition of several new agents. The results are then consolidated and presented to the user. Where it would otherwise require a user to refer to more than one system, a single custom user interface is provided that provides already analysed information.

Following consultation with asset engineers, it was necessary to add the functionality to archive diagnostic information produced by PEDA. This was added to a later version of the system as an additional agent, showing the extensibility of the system.

The system is once again built on the JADE platform, implementing the FIPA specifications. The need to build the system on a robust set of standards was considered necessary for an industrial application. This was not the case with earlier versions [106], where it was found that agents could only run for hours without problems. Following adoption of the now more mature standards and platform, the new version has been shown to be very reliable. It was successfully deployed with Scottish Power for nearly two years [107]. This demonstrates that MAS-based implementations are now robust and adaptable enough for deployment in the power industry.

3.6 Summary

This Chapter has provided a description of multi-agent systems and their capabilities. It has provided an overview of the technology and standards necessary to develop an effective MAS. It concludes with a number of examples of the successful application of such systems to a range of engineering problems.

Chapter 4

Anomaly Detecting Neural Networks for Pulverising Mills

Anomaly detection in pulverising mills can be difficult due to the complexity of the design, and difficulty in capturing sufficient knowledge of the workings of the mill [10] [11]. This often makes using a model-based or knowledge-based approach problematic. Many of the relations between variables are non-linear, further complicating the task of anomaly detection.

The following chapter describes an intelligent algorithm, based on ANNs for the detection of primary air (PA) pressure anomalies in a ball and race-type pulverising mill. This does not require expert knowledge or detailed models of the system. It is shown to be able to detect anomalies where an existing condition monitoring system were unable to detect the presence of an anomaly before failure, and could not prevent an extended outage. It can detect pressure anomalies before an event occurs, allowing it to be avoided, along with the associated costs of repair/replacement and equipment down-time.

4.1 Pulverising Mills for Coal-fired Power Stations

Pulverising mills are responsible for breaking down the coal into a fine powder. The most common types are ball (tube), and ball and race. Pulverised coal behaves much like a fluid, allowing it to be mixed with the PA. This primary air is the main source of air used to mix with and transport the coal to the furnaces, and is generated by a large PA fan. A venturi tube is often placed here in order to monitor the air flow, and

is known as the PA fan venturi. The primary air picks up the pulverised coal, and the combined air and coal enters the furnace where it is burned [10]. The water in the pipes surrounding the furnace is turned into heated steam, which can then drive steam turbines which in turn drive a generator to produce electricity [108]. The layout of a hypothetical generating unit is illustrated in Figure 4.1. The diagram is simplified, and shows a single mill feeding the boiler which in turn provides steam for a single steam turbine. In fact, it is common for a coal-fired power station to have multiple mills per boiler and multiple boilers per turbine set, which would typically consist of a high pressure (HP), intermediate pressure (IP) and several low pressure (LP) steam turbines.

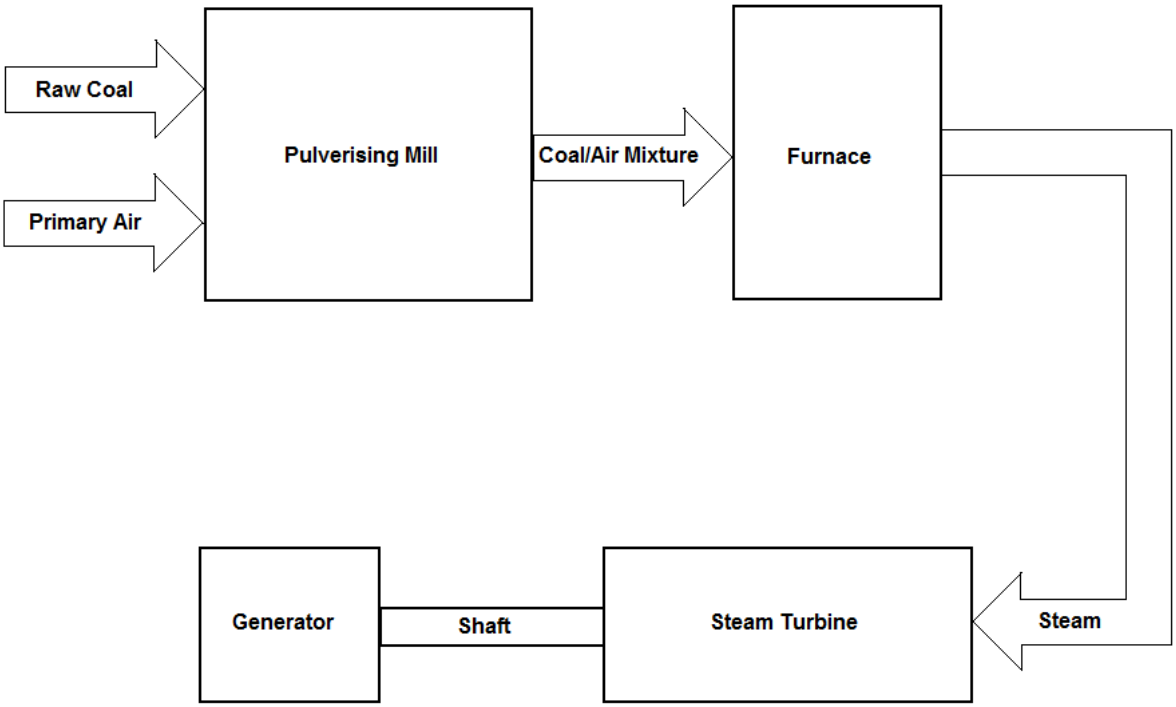

Figure 4.1: Simplified diagram of a coal-fired plant.

4.1.1 Ball Tube Mill

A ball tube mill (often referred to simply as a "ball mill") consists of a rotating cylinder containing many balls made of a hard material [109]. The rotation of the cylinder ensures the balls are kept in a constant state of motion, and they grind down the coal as it

is fed in. If the coal is sufficiently fine then it will be picked up by the airflow passing through the mill, along the axis of rotation. Any coal that is still too coarse is recirculated back to the mill for further grinding [110]. Figure 4.2 shows an example of a ball tube mill.

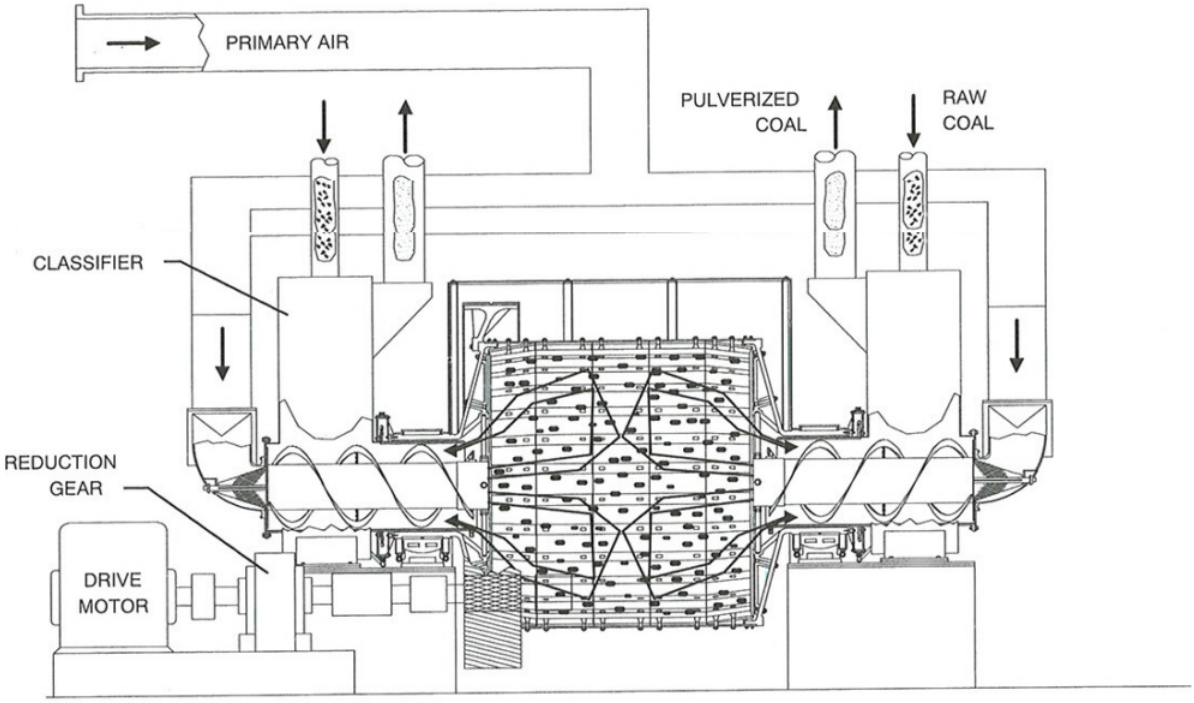

Figure 4.2: The arrangement of a ball and tube-type mill [111]

4.1.2 Ball and Race Mill

A ball and race-type (alternatively known as ball and ring) pulverising mill also uses balls, but additionally utilises an upper and lower "race" to grind the coal down [10]. The balls are arranged in a circle, with the coal entering the centre of the circle. The upper race is also circular and typically spring loaded, and provides pressure on the balls. The lower race is the "bowl" that holds the balls and coal. As the lower race rotates, the balls also rotate and grind the coal. The layout is shown in Figure 4.3. The hot air is passed in and picks up the pulverised coal as it rises towards the outlet. A classifier is also placed below the outlet to ensure that any coal that is not sufficiently

pulverised is returned to the mill.

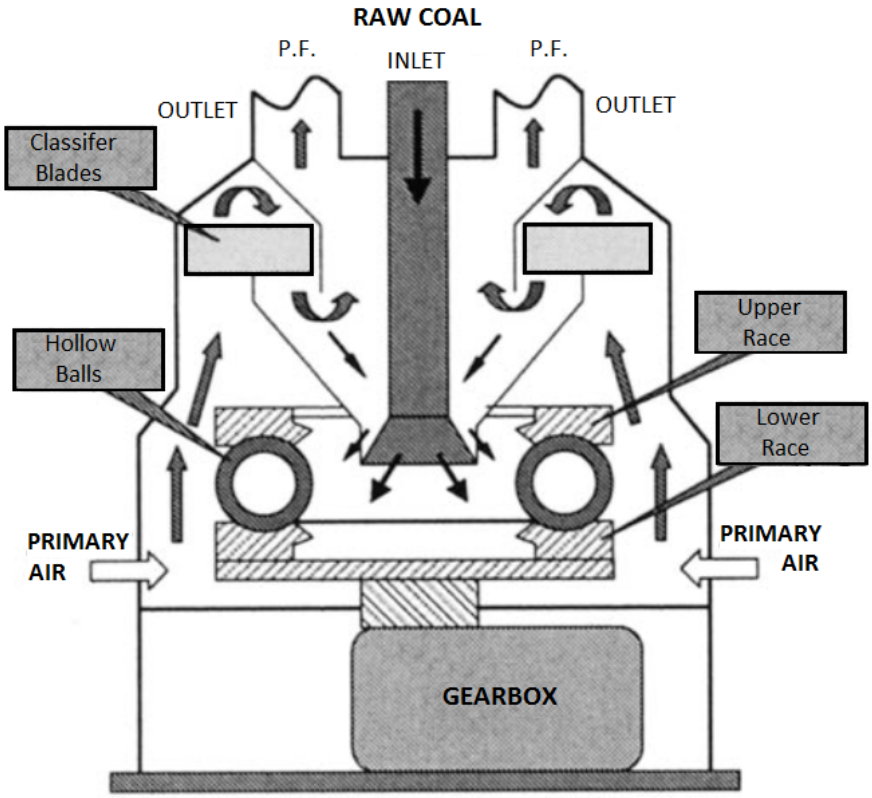

Figure 4.3: Example layout of a ball and race mill [10].

Counter-rotating designs exist, where both the upper and lower race rotate. It is also possible to cascade multiple rings of balls. These features are intended to increase pulverising output and/or capacity.

4.1.3 Review of Pulverising Mill Condition Monitoring

Developing models for pulverising mills is difficult and time-consuming to develop due to the complexity and non-linearity of the system [10] [11]. Variation in design makes the adaptation of existing models difficult, and a utility may need to develop several models to allow monitoring across all mills. Zhang et al. have proposed the use of genetic algorithms to allow an accurate model of a pulverising mill to be developed [10]. By allowing a machine learning algorithm to optimise models, this reduces the man hours necessary to develop the models and makes fleet-wide monitoring of mills

possible.

Most innovation in the field is related to automated controllers rather than true condition monitoring. Control and setting of the parameters of the pulverising mill may be performed through careful monitoring of pulverised fuel mass flow and particulate size [112], but the solutions tend to simplify the control problems to single input-single output (SISO). Chai et al. have successfully developed an approach using ANNs to allow multiple input-multiple output (MIMO) control [113]. Application of fuzzy set theory has also proven successful for control of ball-type mills [11].

Si et al. have proposed a combination of data fusion and radial basis function (RBF) ANNs to monitor the condition of a ball-type pulverising mill [110]. Each of the two trained ANNs produce a condition value - air draft condition and pulverising condition. Fuzzy logic is also proposed, which may use these values as input. They state the difficulties of producing an accurate mathematical model as the prime reason for choosing a machine learning approach. A back-propagation neural network has also been used to model a ball mill [114], with the use of grey relational analysis [115] to reduce the number of necessary input parameters.

While the literature suggests that modelling of pulverising mills is very difficult, there has been some success [109]. Cheetham and Billings propose modelling a pulverising mill by dividing it into three separate multiple input-single output models: PA, temperature and feeder system [116]. Wang et al. suggest that because the parameters used for monitoring a mill are strong coupled, the use of ICA should be applied to allow the most useful features to be extracted for analysis [117] before modelling.

Despite the issues with mathematically modelling a pulverising mill, attempts have been made by accepting a model of compromised accuracy [118]. They cite several failed attempts at a full model, where they are too complex to be practical or are incomplete, as reasons for choosing a sub-optimal solution. The completed model breaks the pulverising mill down into five blocks based on the physical components of the pulveriser.

Upon review, the majority of the literature appears to be primarily focussed on ball mills, and the field is not as developed as those for boiler, steam turbine, or generator monitoring. Modelling of the sub-system of pulverising mills, using a technique capable of capturing the non-linear and complex relationships of the variables, may allow more accurate condition monitoring. This in turn would prevent failures which at present go undetected by the systems currently deployed.

4.2 Plant and Event Description

The data provided by SSE for analysis was from a pulverised coal-fired power station. The pulverising mills were of the ball and race-type. A pressure excursion was known to have occurred in one of the pulverising mills. Each unit of this plant contained several ball and race mills used to pulverise the coal before it is passed to the furnace. The event resulted in the failure of the PA Fan Venturi. The event occurred with little warning; the first indication of a problem was a swing in boiler draught pressures followed by a loud bang in the PA Fan Venturi.

The purpose of the research presented in this chapter was to analyse the data and identify a technique capable of detecting an anomaly in the pulverising mill before the failure occurs. To this end, the data mining methodology outlined in Section 2.1 was followed. Two sets of variables were used, one using the variables used by the existing monitoring system, and an alternative set. The performance of three machine learning algorithms were compared: clustering, in the form of K-means and DBSCAN, and ANNs. Ultimately, a successful algorithm is presented using an appropriate data set and machine learning technique.

4.3 Data Familiarisation and Visualisation

The purpose of this step is to perform a first-pass the data. The general make-up of the data set is considered, and any obvious discrepancies or patterns in the data can be flagged.

Data was received from the PI database in the form of a comma separated value (CSV) file. This included a total of 16 variables, listed in Table 4.1.

The data is in compressed form, only showing new values, along with a time stamp, when the parameter value changes sufficiently.

Several examples of normal shut-down were extracted from operating data and compared to the event. Each extracted set of shut-down data is 40 minutes long. Visual inspection suggested that 30 minutes was sufficient to show the run-down before the shut-down, and only around 10 minutes after were necessary to show the behaviour of the variables shortly after. From this initial inspection of the data, it was possible to determine that, as the anomaly had occurred during shut-down, data from similar operational periods could be focussed on. The appropriate window-size was identified,

Table 4.1: Pulverising Mill Variables

Variable No Description	Units
Mill B Temperature	C
Mill B PA Fan Bias	%
Mill B Tempering Air Act. Posn.	%
Mill B Windbox A Side Pressure	mbar
Mill B Diff. Pressure	mbar
Mill B Windbox B Side Pressure	mbar
Mill B Feeder Rate	%
Mill B Sec Air Posn. A Side	%
Mill B Feeder Amps	%
Mill B Sec Air Posn. B Side	%
Mill B PA Diff. Pressure	mbar
Mill B Inlet Temperature	C
Mill B PA DVC Posn.	%
Load (Generated)	MW
Mill B PA Fan Amps	%
Mill B Amps	%

and several cases of shut-down data could be extracted.

4.4 Data Reduction - Variable Sub-set 1

The first sub-set of variables to be considered were those used by the existing condition monitoring system. These variables were Mill B Temperature, Mill B PA Diff. Pressure, Mill B Feeder Rate, Mill B Amps, and Mill PA Fan Amps. A visual inspection across several cases was carried out, and ICA applied in order to identify anomalous characteristics.

4.4.1 Visual Inspection

A plot of these variables from the time of the event, taken from the event report, is shown in Figure 4.4.

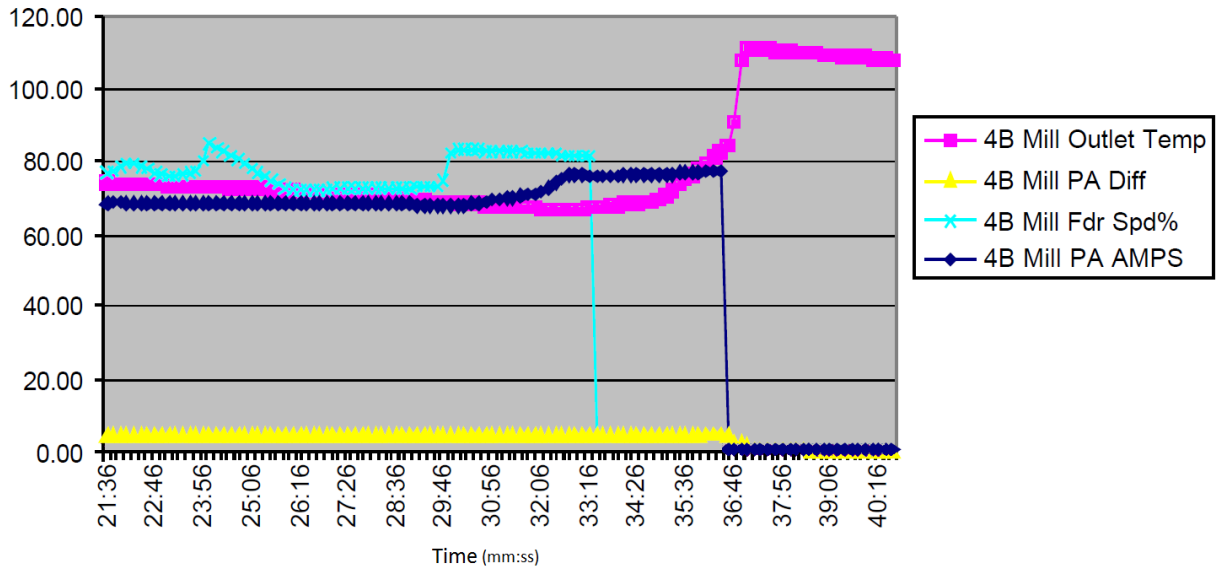

Figure 4.4: Initial Variables from time of the Event.

Inspection of Figure 4.4 shows no unusual behaviour in the run up to the event. The plant was shut down at 31:01, and the feeder speed drops to zero at 33:16. As expected, the PA also shuts down at 36:46, shown by differential pressure and current dropping to zero. However, the outlet temperature showed a large increase at 37:00. This corresponds to the time of a reported audible bang from the PA Fan Venturi.

4.4.2 Application of Independent Component Analysis

ICA was used in order to break the data into subcomponent signals, with the expectation that a single anomalous component may be identifiable and used to detect the anomaly.

Examples of ICA on the event and on a normal shut-down, using the initial variables, are shown in Figure 4.5 to Figure 4.8.

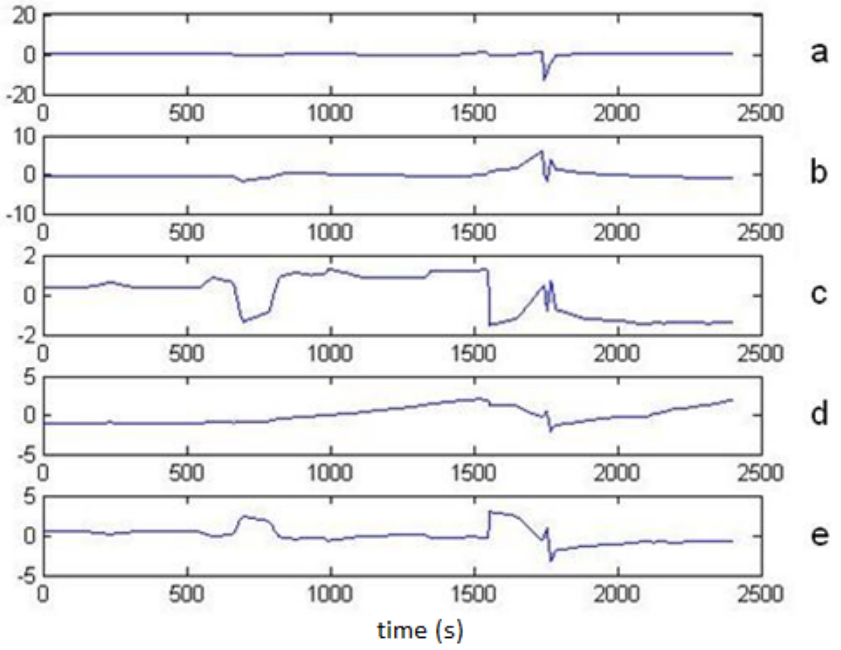

Figure 4.5: Event

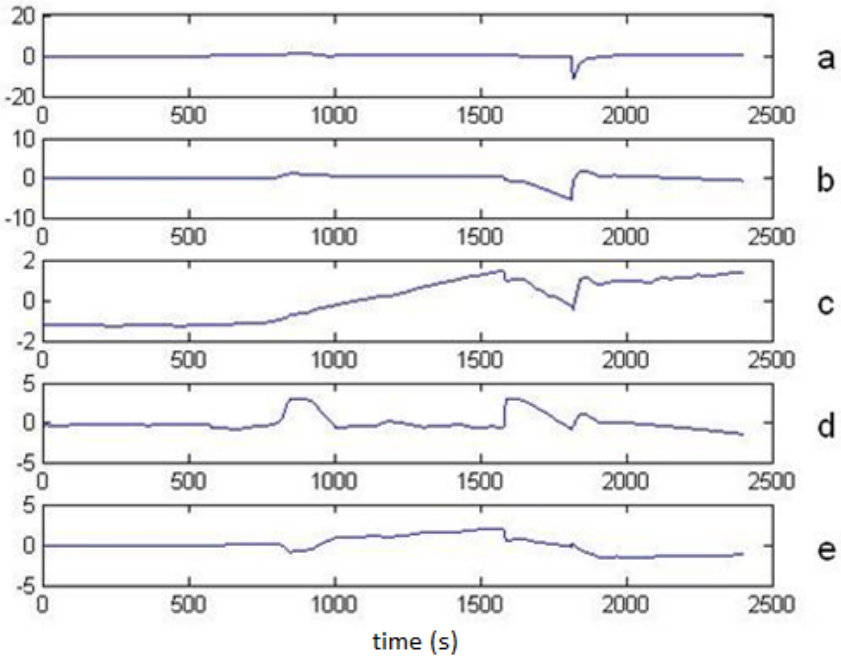

Figure 4.6: Normal Shut-down 1

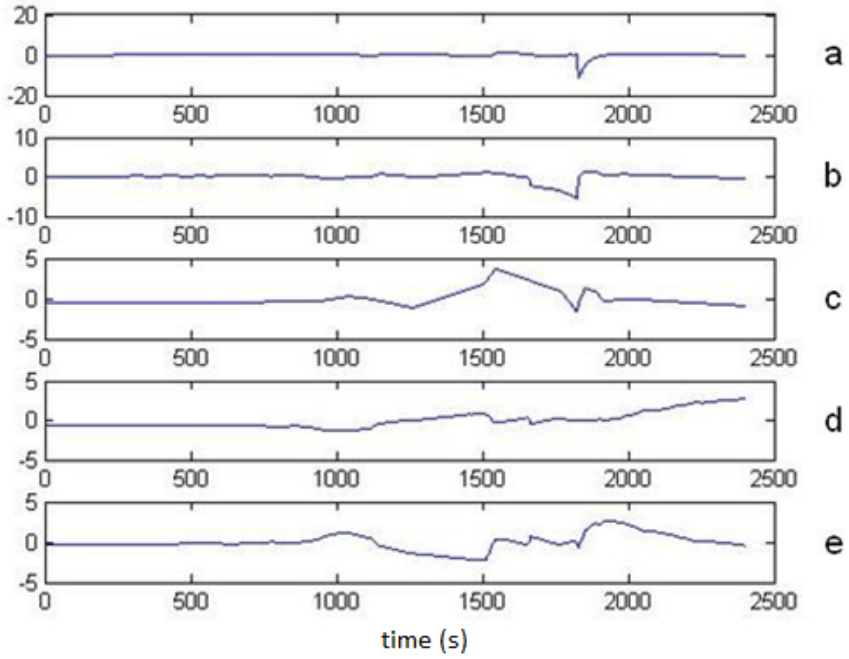

Figure 4.7: Normal Shut-down 2

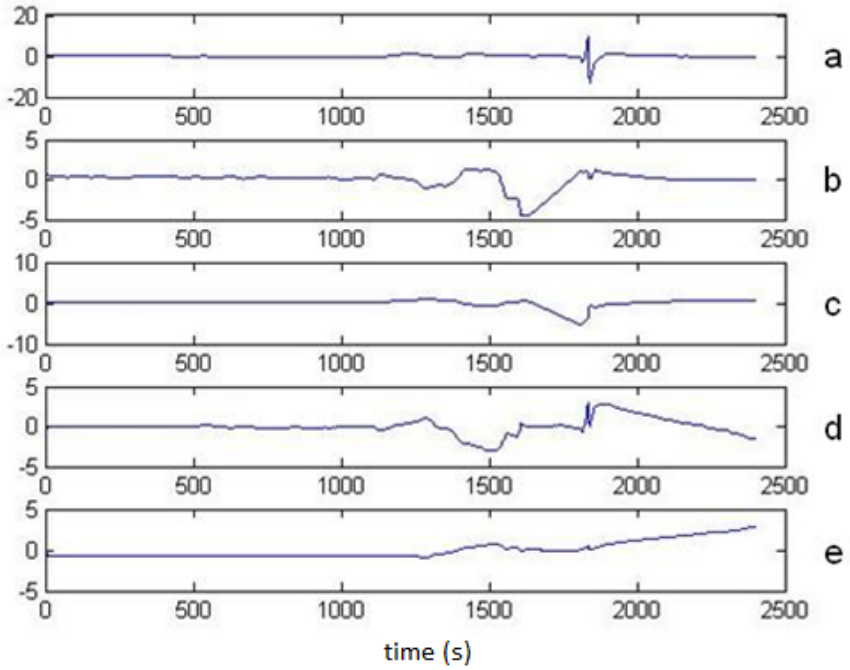

Figure 4.8: Normal Shut-down 3

4.4.3 Interpretation

The components a-e in Figures 4.5 to 4.8 are from ICA analysis of each shut-down, and no physical significance can be ascribed to these. However, it is possible to interpret the features and attempt to match them to features in the orignal data, and compare the components found in different shutdowns in an effort ot identify any discrepencies which may indicate a fault.

Component a is common in 4.5 to Figure 4.7. The same is true of b, although it is inverted in the event (this is due to ICA's inability to determine the sign of components), and shows a "spike" in this and other components, that could be the signs of an anomaly. However Figure 4.8 shows a similar trait in several components (notably a and d) in normal shut-down 3, undermining this "spike" as a potential fault indicator.

The remaining components show more variance between shut-downs. For example Figure 4.5e, taken from the event data, and 4.6d, from normal shut-down data, appear to be the same component. However, Figure 4.7 is also from a normal shut-down and does not show a similar component. Therefore, while ICA tended to find similar components

in the data, it did not provide a clear and consistent differentiation between normal and erroneous data, and often showed as much variance between normal operating datasets as when compared to the event.

The results suggest that ICA was unable to identify a single component which would identify the event data as anomalous. The failure of the existing system to identify an anomaly, and the absence of indicators from visual inspection, and of anomalous components following ICA suggest that these variables are not suitable to detect a PA pressure anomaly.

4.5 Data Reduction - Variable Sub-set 2

An alternative variable sub-set was sought. Applicable literature was reviewed to guide the variable selection. Cheetham and Billings [116] suggest the mill variables be split into three variable sets: PA, Outlet Temperature and Feeder. Following splitting the system up in this way, they develop a mathematical model of each sub-system. The approach to splitting the mill variables is emulated in the approach taken here but due to the abundance of data present, and a consensus in the literature that an accurate mathematical model is difficult to achieve [10] [11], data-based approaches were sought instead.

4.5.1 Visual Inspection

The pressure variables were compared between the event and normal shut-down data.

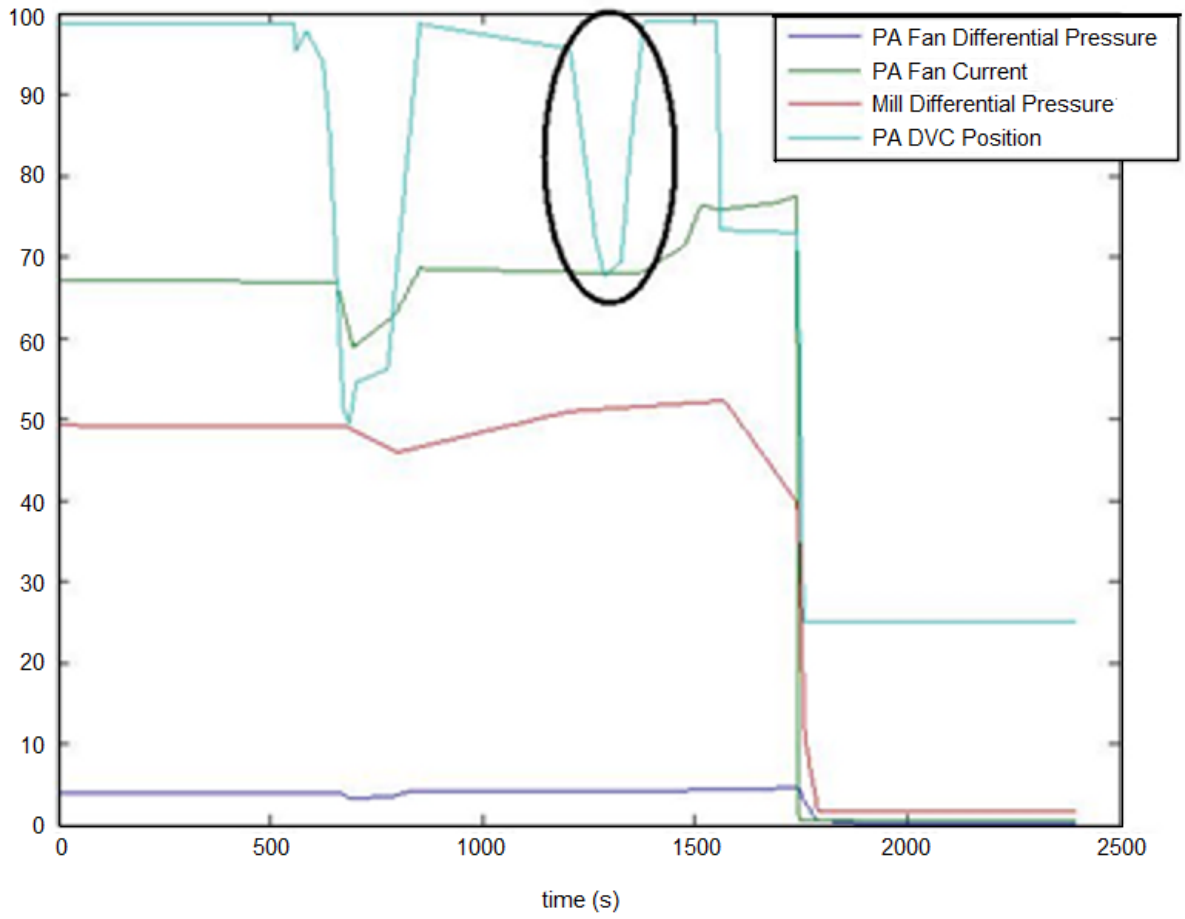

Figure 4.9: The pressure variables during event. The dip in PA DVC position is circled. Of importance is the lack of change in the other variables, which elsewhere are highly correlated.

It was found that the PA differential voltage control (DVC) Position (input) variable experienced a dip a few minutes before the event (shown in Figure 4.9). While this is not of itself unusual, it is unusual that this dip does not affect the output variables, which are otherwise strongly correlated with the input variable in all other shut-downs.

4.5.2 Interpretation

The presence of an apparent deviation from normal behaviour suggests that this set of variables may offer a means to detect the appropriate type of anomaly. For this reason, this variable sub-set was chosen for the comparison of machine learning algorithms.

4.6 Technique Selection - Clustering

K-means and DBSCAN were chosen in order to identify the possibility of using a clustering-based approach for anomaly detection.

4.6.1 Methodology

Both clustering techniques were used to group high-dimensional data points in an attempt to see a pattern in the system's movement through states, with the intention being to see entry into a recognisable "anomaly" state in the lead-up to the event.

The K-means algorithm was tried several times, manually setting cluster numbers from two to six, but regardless of the number of clusters no clear indication of anomalous behaviour was identified. DBSCAN was repeatedly tested with several different ϵ values (0.1 to 0.9) and minimum cluster size values (2 to 10). All data consisted of the 30 minutes leading up to a shut-down, and 10 minutes immediately following.

4.6.2 Results

Figure 4.10 shows an example comparison between the event and normal shut-down, using X-means, which automatically calculated four clusters before performing K-means, to cluster the event report variables. The colours show clusters, which may be considered as states.

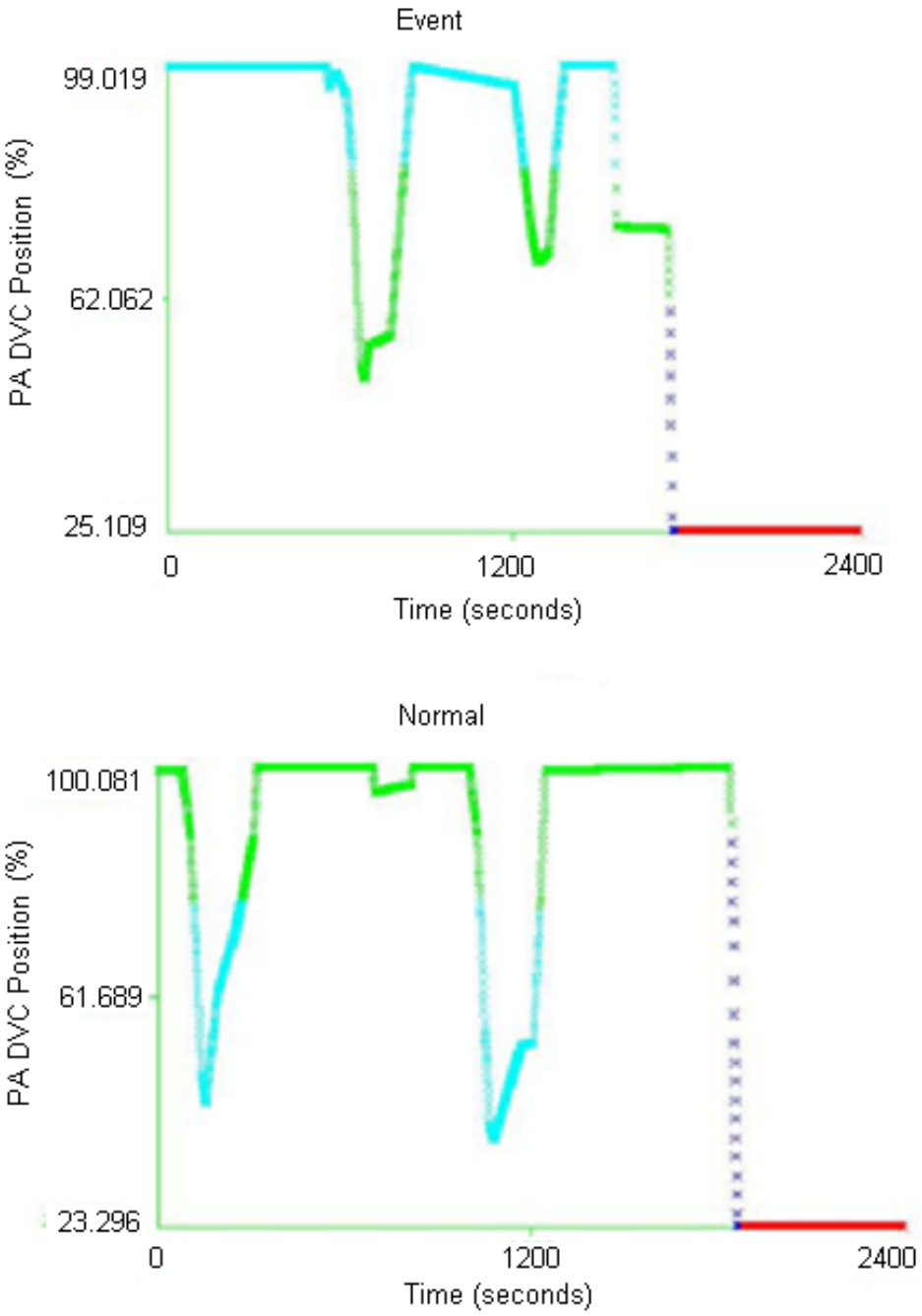

Figure 4.10: PA DVC Position, with colours representing clusters. Event data above, a normal shut-down on the bottom.

The apparent switching of colours of green and blue between the plots is due to the way K-means and similar algorithms calculate initial cluster centres at random and the way in which colours are assigned to those centres during visualisation: not due to some difference in state. It is the behaviour of the plot and the transition between states that is important. Both plots show two similar dips and state transitions leading up to shut-down, before entering a transition state and finally a shut-down state.

The behaviour shown in Figure 4.10 is typical of that seen using other K values, and of that shown by the DBSCAN algorithm.

4.6.3 Interpretation

Despite the presence of the "dip" prior to failure, and the presence of an associated state transition, there is no obvious "anomaly" state that is shown only in the event. The presence of similar behaviour in normal data, suggests that the clustering algorithms are unsuitable for distinguishing anomalous behaviour as required.

4.7 Technique Selection - Artificial Neural Networks

Back-propagation artificial neural networks were selected based on their successful use in several other applications [119] [103].

4.7.1 Methodology

The ANN was trained on the pressure variables (Variable Set 2): PA Fan Current, PA Differential Pressure and Mill Differential Pressure, using PA DVC Position as a numeric classifier. This allows the neural network to learn the relative relationships between the four variables. During the testing phase it was provided with the three output pressure variables, and predicts the PA DVC Position. This can then be compared to the actual value in order to determine how close the values are to normal behaviour.

Data was tested using the Waikato Environment for Knowledge Analysis (WEKA) data mining tool using the MLP classifier function, with three inputs, a single hidden layer with two nodes, and a single output. As all values are taken from the same time sample, the network will not predict based on previous values, and predicts the PA DVC Position based only on the current values of the pressure variables. The network was trained on four samples of normal shut-down data. The test data consisted of two

further normal shut-downs not included in the training set, and the shut-down containing the event.

4.7.2 Results

Figures 4.11 and 4.12 show the residuals of normal shut-downs, while Figure 4.13 shows the event and its correspondingly less accurate predictions.

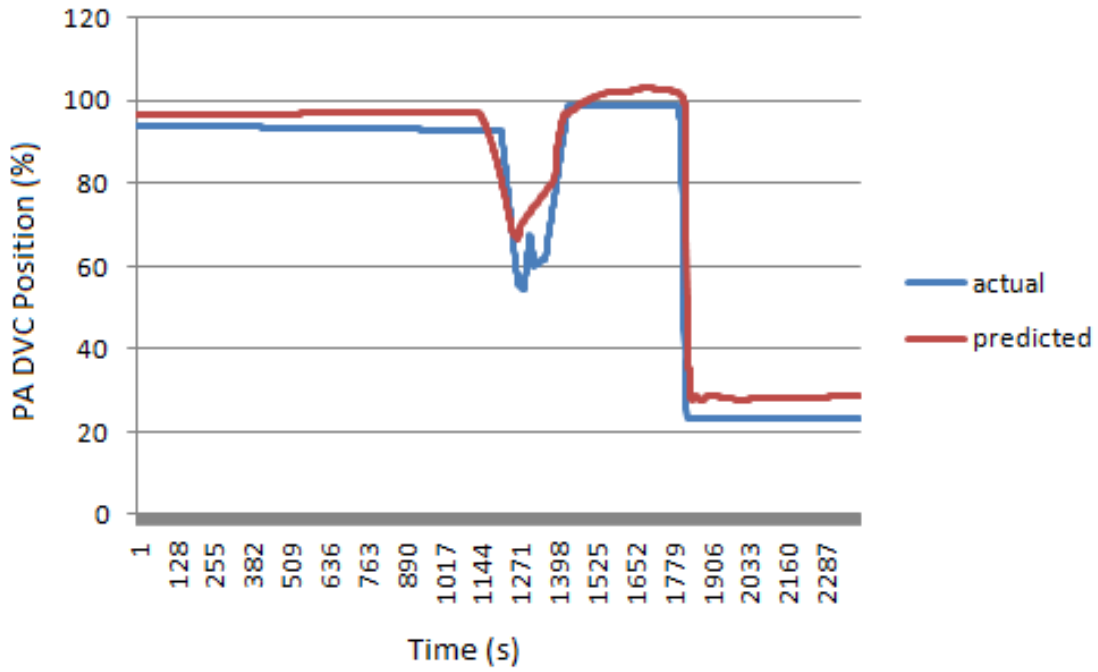

Figure 4.11: Actual and Predicted Values for Normal Test Set 1.

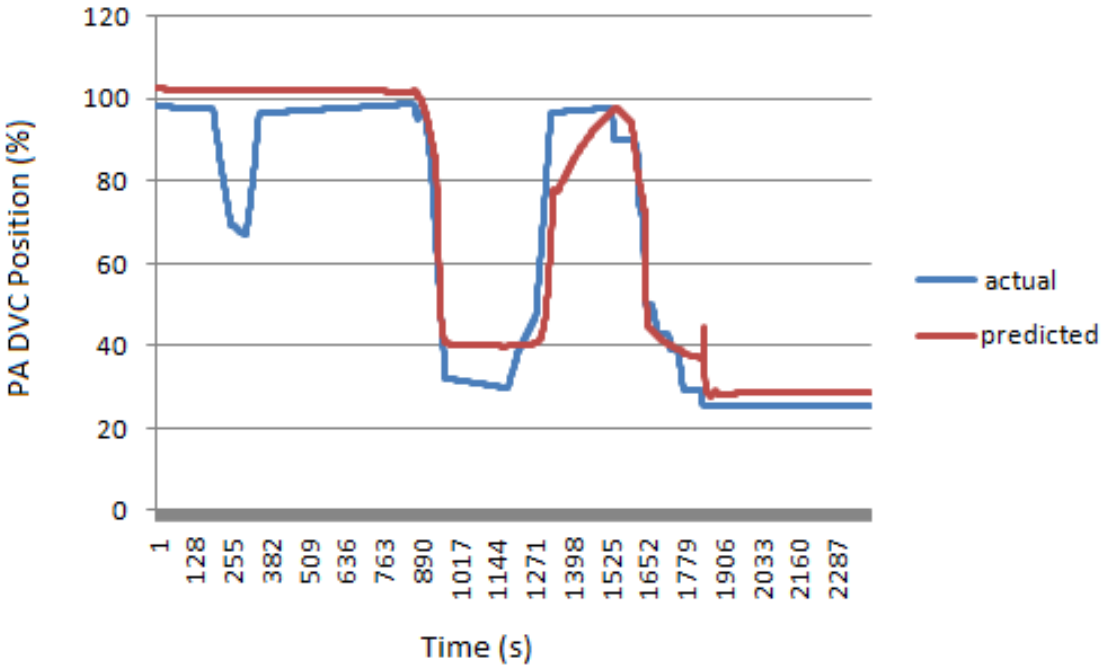

Figure 4.12: Actual and Predicted Values for Normal Test Set 2.

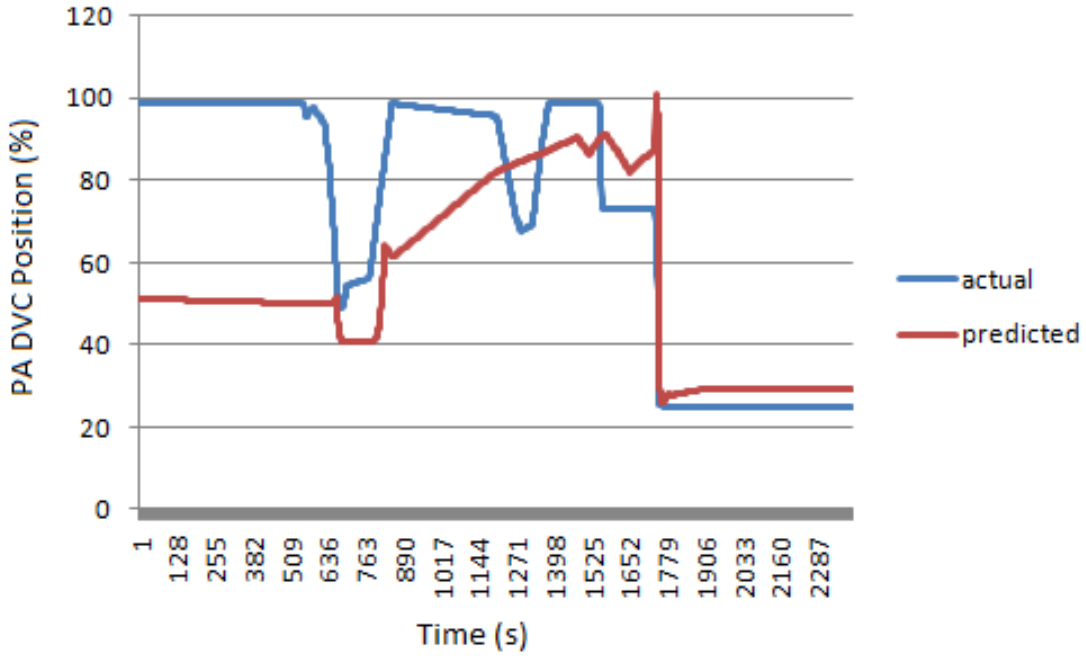

Figure 4.13: Actual and Predicted Values for the Event.

Several statistical metrics following testing of the shut-down examples are, illustrated in Table 2. This several measures of accuracy, which provide further insight into the accuracy of prediction.

Table 4.2: Comparison of Predicted and Actual Values for 40 Minute Shut-down datasets

	Correlation coefficient	**Mean absolute error**	**Root mean squared error**	**Relative absolute error**	**Root relative squared error**
Training 1	0.986	5.5356	7.5703	17.03%	22.78%
Training 2	0.9775	5.1406	7.788	15.95%	23.10%
Training 3	0.9758	6.6506	8.5408	21.02%	25.85%
Training 4	0.9854	6.379	7.0988	19.13%	20.82%
Normal 1	0.9877	4.6823	6.4321	15.16%	20.01%
Normal 2	0.9613	6.817	9.7118	22.05%	30.23%
Event	0.6752	21.7442	28.0293	73.15%	88.19%

4.7.3 Interpretation

It is possible to simply output the residual values between predicted and normal data, but in order to provide an algorithm applicable across asset types a general unitless metric was sought. The Pearson correlation coefficient was chosen as it shows the strength and direction of a linear relationship between two random variables [120], providing a value between negative one and positive one, where zero is no correlation and positive or negative one is full correlation.

Comparing sets of 40 minutes containing shut-down data showed a marked difference in correlation coefficient between all the normal operating data and the event dataset. While all normal sets returned a correlation coefficient of above 0.96, the event has a correlation of only 0.68, as shown in Table 4.2. A preliminary threshold of 0.8 was selected for anomaly detection.

The results suggest that the ANN approach can detect the pressure anomaly related to this fault. However, the event data also included 10 minutes of data from after the event. This potential delay is not acceptable for this application, due to the anomaly (4.9) occurring only around five minutes before the event. For this reason, further refinement of the technique was required to make it applicable to on-line operation.

4.7.4 Refinement

Linear interpolation was used to convert the compressed data into minute-long sections, with a residual generated every second. In order to prevent noise in the data from causing false positives, the last three minutes of data are considered and an average of their correlation coefficients taken. Three minutes was chosen to increase noise resistance but also, due to the short time scale, keep delay low in order to ensure timely detection of the anomaly and to allow engineers to react to the alarm. Data that shows a low correlation coefficient between predicted and actual values but is not supported by surrounding data, as shown by the average with surrounding values, is considered a ”Potential Anomaly”. Data that is consistent with the surrounding values is considered a “Confirmed Anomaly”.

A random fluctuation in readings, perhaps caused by noise or poor generalisation of the ANN for certain multivariate values, which would result in a temporarily low correlation coefficient, would be considered an anomaly. But, when compared to the previous two correlation coefficients it would be found that they show no evidence of an anomaly. As a result, this would be considered a “Potential Anomaly”, as it may be an anomaly, but surrounding values suggest that is more likely to be temporary noise or transient. If, however, the previous values also have a low correlation coefficient, this would reduce the average correlation and these values would ”Confirm” that some kind of persistent anomaly is present, resulting in a “Confirmed Anomaly”.

This technique was applied to the test cases successfully. It can correctly identify a pressure event during shut-down up to four minutes before equipment failure. There were no cases of false-alarms for “Confirmed Anomalies”. This suggests this technique is the superior choice out of the considered machine learning algorithms.

4.7.5 Further Testing

A Java application was created to read directly from compressed data in the form of a CSV file and use the trained ANN to retrieve a correlation coefficient for each one minute dataset.

4.14 shows a comparison of the results, using averaging over three minute windows, between the event and normal shut-down data. On this scale, shut-down occurred at around 20 minutes for both, and the event occurred at 17 minutes. This shows a large dip in the correlation coefficient in the lead up to the event. Initial testing showed the

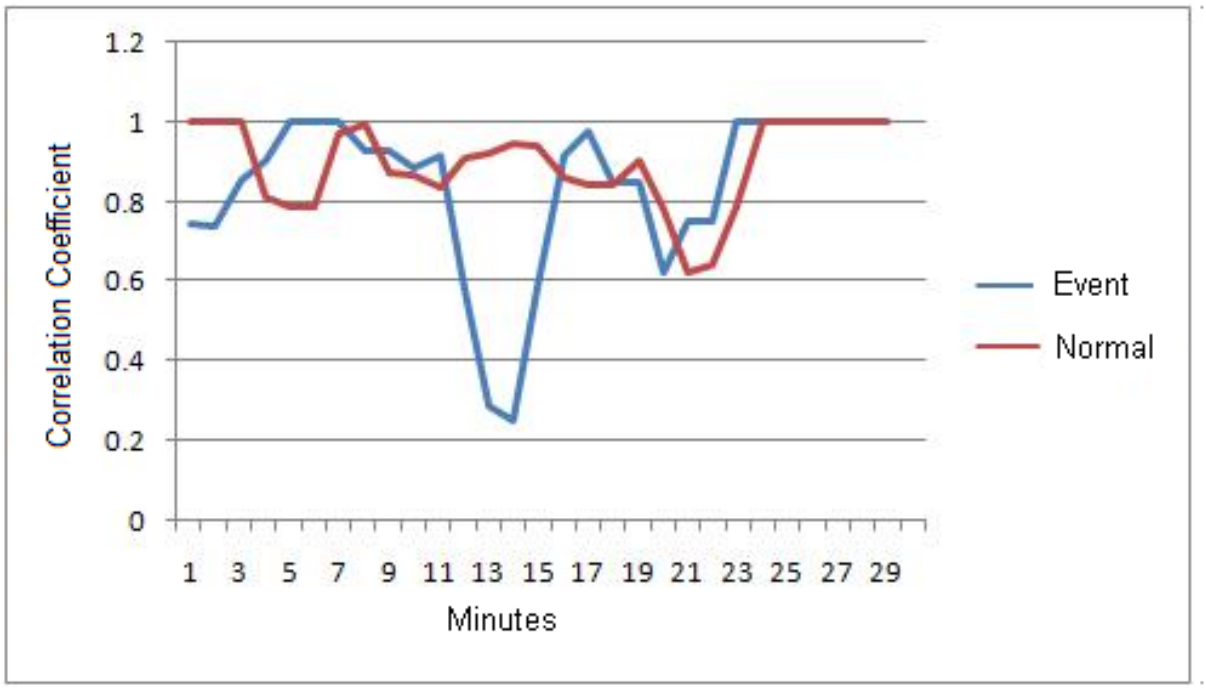

Figure 4.14: Comparison of ANN output, the event in blue and a normal shut-down in red.

shorter one minute window tended to be more vulnerable to noise. Figure 4.14 shows the normal shut-down contained two dips below the intended 0.8 threshold value. Because of this, the threshold for an anomaly was eventually set at a correlation coefficient of 0.5. The confirmation range was set to the current and previous two coefficients. This was performed for all previously referenced datasets, with only the event data showing potential anomalies at 12 and 19 minutes, and a confirmed anomaly at 13 minutes: four minutes before the event.

4.8 Conclusion

This chapter has considered several machine learning techniques, with variable and technique selection driven by a data mining methodology. The contribution of this chapter is an ANN-based anomaly detection algorithm for pressure variables in ball and race pulverising mills. It does not require complex models of the system or extensive expert knowledge, while still capturing the complex and non-linear relationships between the variables.

The algorithm proposed in this chapter has been successfully tested with several data sets. The algorithm provides the capability to correctly identify a pressure event, previously undetectable to existing condition monitoring systems.

Chapter 5

Gas Turbine Condition Monitoring

The following chapter aims to present methods for the condition monitoring of gas turbines, in the absence of high frequency data, while still modelling some aspect of the dynamics of the system. Two methods are presented, and both utilise relatively low-frequency data (as little as one sample every 10 minutes). Building on the experience and information gained from applying them to the pulverising mills in Chapter 4, the first is ANN-based, this time focussed on exhaust gas temperature anomalies and using time-delayed values. Results have proven this method's capability for detection of faults in advance of failure, with potential for application to other fault categories.

The second method proposes HMMs for modelling of GT subsystems, with results of particular note from tests on compressor and combustion data. Two distinct types of HMM are compared, with both, and in particular the MHMM, showing suitability for both anomaly detection and machine degradation tracking. Results suggest that the HMMs are capable of modelling the dynamics of the turbine, despite the absence of high-frequency data.

Following a brief review of gas turbines (Section 5.1) and the current monitoring systems available (Section 5.2), Section 5.3 describes the plant and available data. Section 5.4 outlines a neural network-based approach that allows detection of exhaust gas anomalies, using residual monitoring. Section 5.5 shows that hidden Markov models, both discrete and continuous, may be utilised to monitor the compressor of an in service gas turbine. Section 5.6 provides a direct comparison to an existing condition monitoring system, and shows that hidden Markov models are superior for the monitoring of combustion within a GT. The chapter concludes with a number of proposed techniques to allow the automation of the HMM-based methods by using automated analysis of

model outputs (Section 5.8).

5.1 Gas Turbines

GTs are a wide and diverse family of machines, although most commonly known for propulsion or electrical power generation. Fundamentally, all GTs take in air through an inlet, compress it, burn it with a combustible fuel, and output the exhaust gas through an outlet, which may or may not be designed to provide thrust. Regardless of the particular application or type of turbine, the core of a gas turbine consists of the following components: the compressor(s), combustors, shaft(s) and turbine(s) [121].

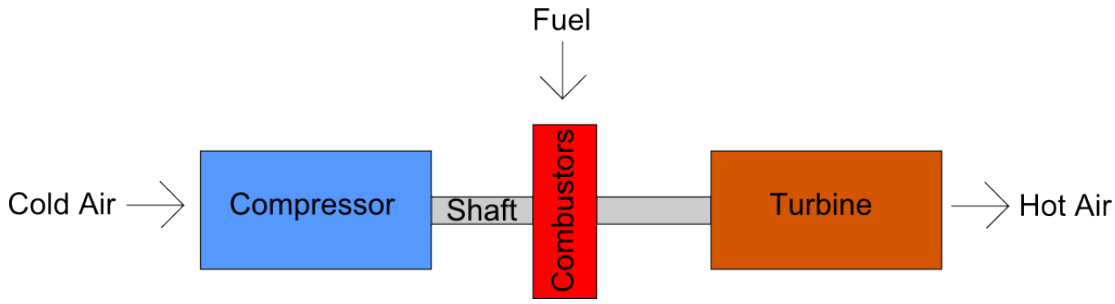

Figure 5.1: The main components of a simple (single shaft) gas turbine

5.1.1 Types of Gas Turbine

Aero engines are now commonly turbofans, and include a fan that provides a component of the thrust, powered by a shaft connecting it to a turbine. A bypass area around the engine core allows this accelerated air to pass by the rest of the GT and mix with the core exhaust. An aero-engine can have 1-3 shafts, operating concentrically, with each having a turbine at one end and a compressor and/or fan at the other. A turbofan derives thrust from a combination of the bypass air accelerated by the fan and the hot gas from the core exhaust.

Generation is possible using aero-derived engines. These are aero-engines with the exhaust routed to an additional turbine, called a power turbine. The power turbine is connected via a separate shaft to the generator.

For large scale generation, the large and heavy-duty frame-type gas turbine is utilised, which does not derive thrust from the hot exhaust gases. They are typically not bypass

engines, meaning all air passes through the core. They also tend to be single shaft, with the shaft connected to a generator to produce power.

Any reference to a gas turbine in this thesis, unless explicitly stated otherwise, is referring to a large, single-shaft, non-bypass, frame-type turbine.

5.1.2 Compressor

The purpose of the compressor is to compress the air drawn from the inlet in order to allow the maximum amount of fuel to be burned in the combustor(s). There are two main types of compressor: centrifugal-flow and axial-flow [122]. Axial-flow compressors have become the typical choice for most large gas turbines. A single stage of an axial-flow compressor consists of a set of rotating blades called a rotor, followed by a diffusing set of stationary blades, called a stator. Each compressor stage compresses the air by first increasing the velocity of the air using the rotor. This velocity increase is then converted into an increase in pressure through the stator blades acting as a stationary diffuser. An additional stationary set of vanes may be placed before the first stage in order to guide air from the inlet into the compressor at the optimal angle. These are called inlet guide vanes (IGVs), and may be variable in order to allow the turbine to increase the range of conditions over which the it may operate [121].

Turbine efficiency and output is related to the compression ratio, provided firing temperature in the combustors is also increased. Pressure ratios have therefore increased steadily, with current frame-type turbines having a typical compression ratio of around 35:1. While a single stage has a typical compression ratio of between 1.1:1 and 1.4:1, by staging several of these sets of blades one after the other, it is possible to achieve the kind of overall compression ratios found in modern turbines. This typically means 15+ compressor stages in a modern gas turbine [121].

A common source of degradation is compressor fouling [9]. This is caused by contaminants in the inlet air building up on the compressor blades. This can be mitigated by both on and off-line water washing [123], among other techniques. All maintenance must be scheduled, and for this reason it is advantageous to the utility to be able to model the degradation of turbine performance [7]. Compressor efficiency is often used as an indicator of compressor fouling, but this simple measure cannot be used to detect unexpected degradation in other parts of the turbine. In addition, compressor efficiency is dependent on inlet air temperature, meaning that variances in ambient temperatures may compromise the reliability of this measure [121].

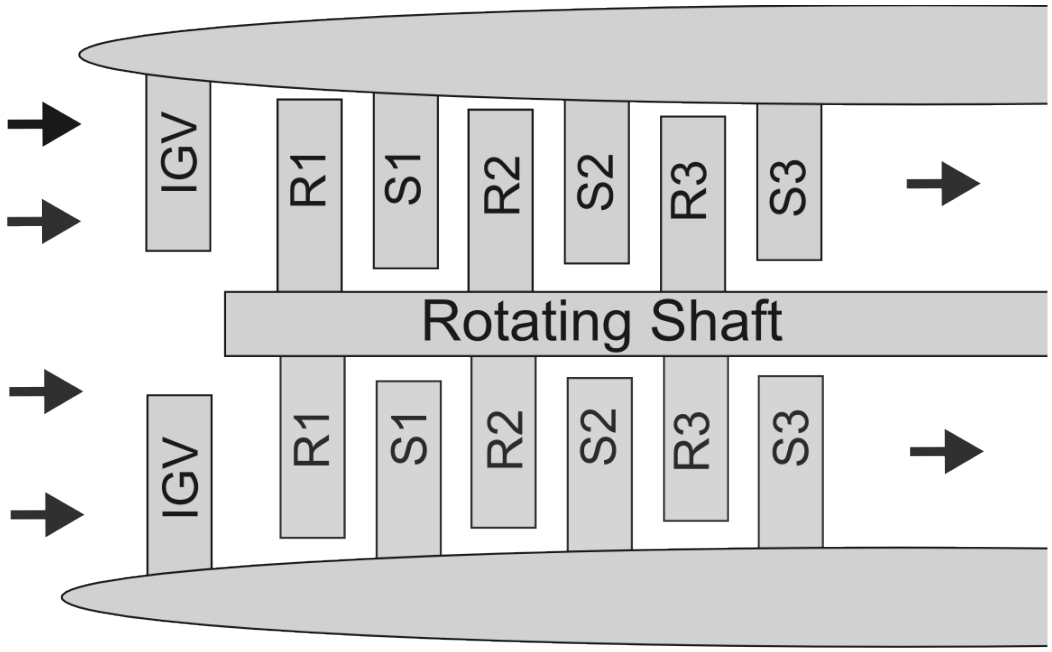

Figure 5.2: Simplified layout of a 3-stage compressor

5.1.3 Combustion

The process of combustion is perhaps the most important part of a gas turbine. It is the means by which energy is transferred to the system. This process is performed by the combustors, which burn the GT's fuel with the compressed air from the compressor. These come in many forms, with the most common being can-type, annular, and cannular(or can-annular).

Can-type combustors consist of a can shaped casing and liner, with one or more fuel injectors. Several are used, arranged around the circumference of the turbine. An annular combustor consists of a single large casing and liner around the entire circumference or the turbine. As there is essentially a single combustion zone around the whole circumference annular designs provide several advantages including more even combustion, less pressure drop, and are lighter and smaller. However, annular combustors tend to be more difficult to develop and maintain [121]. Cannular designs consist of separate cans but are contained within an annular casing. Cross-fire tubes are used to improve the spread of combustion across the cans. These have a measure of the advantages of annular designs, but are easier to test and develop. While annular designs

are still lighter and require less cooling, making them ideal for high-performance jet engines, cannular designs are typically used for large frame type turbines.

Combustion cans can also be reverse flow or straight through flow, referring to how the gas is directed through the combustion can. Straight through flow combustors allow the frontal area profile of the turbine to be reduced, making this type ideal for aircraft applications. However, heavy duty frame type turbines tend to use reverse-flow combustors, with advantages including a shorter engine and shaft [121].

5.1.4 Shaft

The shaft is used to connect a turbine to whatever component it is driving. Depending on the application this may be a compressor and/or generator, or some other piece of machinery. While it is possible for aero and aero-derivative engines to have multiple concentric shafts, each connecting a high, intermediate or low pressure turbine to its counterpart compressor/fan, a typical frame type GT has only one. An aero-derivative engine will also require an separate shaft to connect its power turbine to the generator, where a frame type turbine is connected to both the compressor and generator via the same shaft [124].

5.1.5 Turbine

The turbine section of a GT can have two purposes. It may be to drive the compressor, or to drive a generator. In either case, the turbine is attached to the device it is driving by a rotating shaft [124]. Turbines are in many ways like a compressor in reverse. Like compressors, they come in two types, axial-flow, and radial-inflow which are the counterparts of the axial-flow and centrifugal-flow compressors respectively. Radial-inflow turbines are very rare, especially for large frame type GTs, which are normally axial-flow [121].

In the case of aircraft propulsion the turbines are designed to extract only enough energy to drive the compressor, with the excess energy providing thrust. A power turbine is designed to drive a generator, and therefore is designed to extract the maximum amount of energy to allow its conversion to electricity [124]. A power turbine, generator, and connecting shaft can be added to existing aero engines to create a derivative engine capable of power generation. In the case of large frame type GTs the turbine is designed to drive both the compressor and the generator along the same shaft. If used

in a CCGT plant, it is also possible that a steam turbine may also drive the same shaft [121].

5.2 Review of GT Condition Monitoring Techniques

The utility has several plants containing gas turbines, either for peaking load, or as part of a CCGT plant. Gas turbine monitoring techniques show a great deal of diversity. Maalouf et al. [125] state the variation in gas turbines monitoring is as a result of differences between Original Equipment Manufacturers (OEMs) and the original application of the turbine - either aero-derivative or dedicated power generation. This is also due to the large number of potential sources of failure [1].

Often a plant-wide database may exist consisting of relatively low frequency data, with sampling rates of several seconds to minutes. Data from this source can be useful, but it is difficult to capture the dynamic behaviour of a gas turbine rotating at high speed without a high sampling rate and low delay [126]. When the data sampling rate is reduced to below the Nyquist Rate, such as in the case of general plant databases, many approaches become inapplicable [127]. This often results in a second system being employed using higher frequency readings to better capture the dynamic behaviour of the turbine. However, this will be only for a limited number of specific variables, typically vibration, and requires separate analysis from the other data.

Aero engines may have vibration condition monitoring systems present, but due to the practices of the aerospace industry, band pass filters are used to filter out virtually everything except the x1 vibration signal. This is because it is intended only to give a pilot a "GO" or "NO GO" result, and not to predict time to failure, which can be determined during ground-based maintenance. There are automated approaches involving signal analysis of vibration or other high-frequency data using Fourier or Wavelet transforms of the captured data that are applicable to power-industry gas turbines. Examples of such systems include PolyProtech [128] and Symbolic Dynamic Filtering [129].

Alternative approaches include building a probabilistic model of possible faults. Poncet et al [43] take this approach, by focussing on models that predict the likelihood of particular faults in the compressor and turbine, based on available symptoms. This is, however, extremely complicated, and would require extensive knowledge of the turbine and the operation and degradation of both components during each fault. Other techniques such as similarity-based modelling (SBM) have been shown to be effective

by modelling the similarity of data vectors to existing training examples [130] [131].

Rather than diagnosing and classifying the particular type of fault, approaches to GT's often focus on an anomaly detection approach, where the primary goal is the detection of any deviation from normal behaviour. This is often the most suitable avenue for condition monitoring of high integrity systems [132]. For GTs, where faults are relatively rare, a multi-class fault classification approach is not always possible due to the lack of data for even a sub-set of possible faults. It is therefore useful to instead model the normal behaviour of the GT, of which there is usually an abundance of data, and raise alarms when the behaviour deviates sufficiently from normal.

Mathematical models may be applied, but these tend to be very complex. The OEM may have models, but will not necessarily be willing to share them. While models can be derived, this is not a trivial process. Although acceptable models for heavy-duty gas turbines have been developed [133], these are still based on a particular set of turbines. Application to other turbines typically requires derivation of several model parameters [134].

Ultimately, some kind of machine learning approach is more appropriate in a case where knowledge of the equipment is limited, but data is abundant. The DADICC system [6], which utilises ANN as models for components of a CCGT, shows the application of such a technique. While the effectiveness of this approach has been demonstrated through deployment on an Iberdrola CCGT plant, it is stated that the choice of variables is vital to the success of such models. This strongly suggests the need to use a structured data mining methodology (discussed in section 2.1).

Another example, the ND Tool [135] is a fleet-wide application designed to learn the behaviour of a plant using entirely data-driven techniques. The example uses a form of K-means clustering, and can be applied to any set of variables selected by the user. Using a training dataset based on normal operation, the clusters for normal behaviour are learned and stored. During testing the Euclidean distance between the actual values and the cluster centres is calculated, and used as a measure of normality.

In an effort to capture the underlying state of other types of dynamic systems HMMs have been applied [136] successfully. HMMs have been shown to be effective for anomaly detection of other electrical assets such as transformers [5]. This technique arose from the fundamental work undertaken by Fox et al [56]. While mechanical faults have been shown to be detectable with HMMs [137], in particular continuous HMMs, for a full diagnostic capability it is necessary to monitor the whole system, not just the mechan-

ical or rotating properties. HMMs have proven themselves, with several variations of HMMs applied to rotating machinery [42] [138] [139]. However, none of these have been applied to the full spread of low-frequency data available to power GTs.

Many techniques have found their way into composite condition monitoring systems. Examples include AMODIS [135], which includes tools such as the previously mentioned ND Tool for monitoring entire plants, and TIGER [140] which is focused specifically at GT monitoring. Regardless of the scope of the assets focused on, systems such as these typically cover the processes of data acquisition, pre-processing, anomaly detection, fault isolation and fault diagnosis. Systems may apply only a single algorithm to each task, or may combine several in an effort to minimize the weaknesses of individual techniques. In the case of more specialized systems like TIGER, the turbine may be divided into subsystems of related variables. Such systems are extensible, with additional algorithms being added in or existing ones updated or replaced. None of these systems use HMMs or model the dynamics of the system as proposed in this thesis. However, their extensibility and modularity are considered a requirement for any solution proposed in this thesis.

5.2.1 TIGER

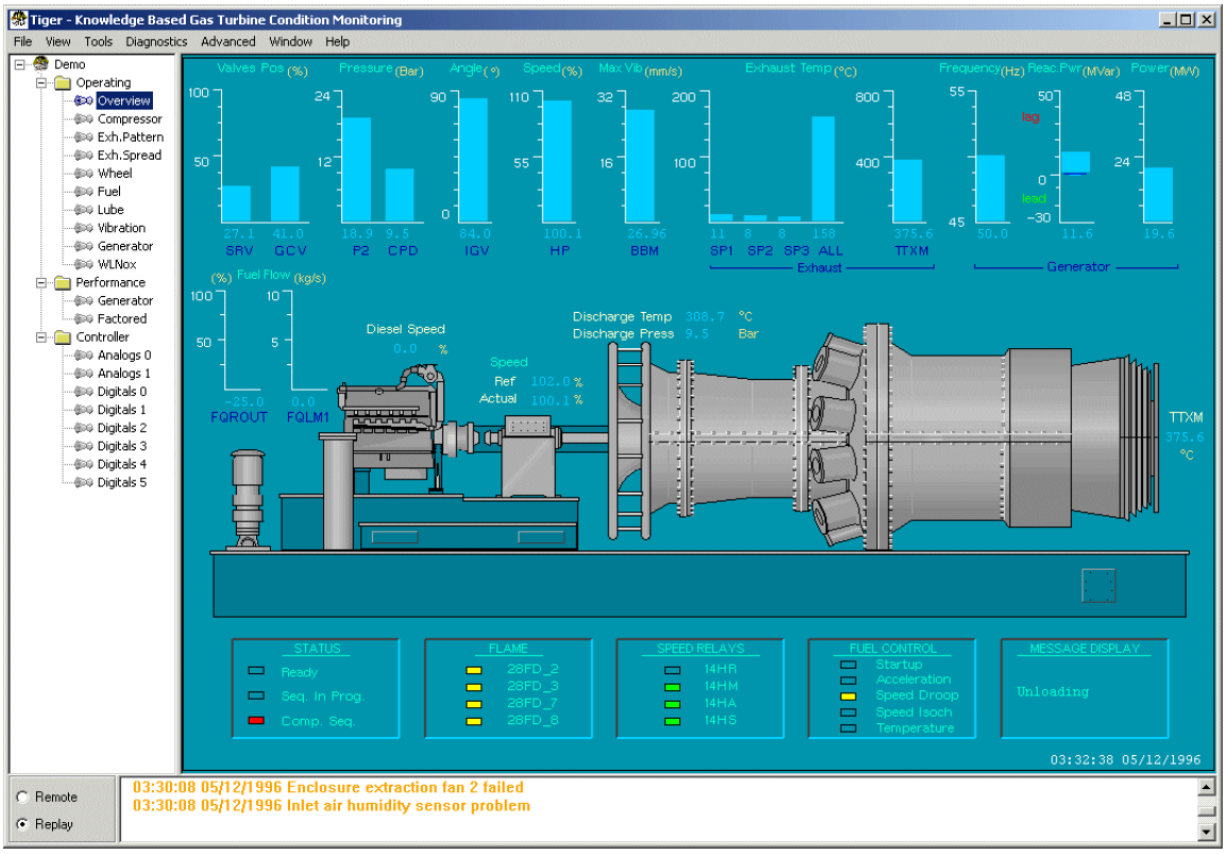

Figure 5.3: The TIGER turbine overview screen [140].

TIGER [140] is a knowledge based gas and steam turbine condition monitoring system developed by Turbine Services [13]. Figure 5.3 shows the TIGER overview screen, with a gas turbo-generator unit and diesel generator, and several important readings.

The gas turbine is split into five major components [140]:

1. compressor
2. exhaust temperature spread
3. vibration measurements
4. fuel system
5. wheelspace thermocouples

These components correspond roughly to those used by the utility's existing condition monitoring system, and the models used for the proposed MAS (See Chapter 6).

The diagnosis system consists of three tools, coordinated by a fault manager [141]. These are CAusal ENgine (CA-EN), KHEOPS and Indexed Time Tables (IxTeT).

CA-EN [142] This implements a form of causal modelling. It was designed to overcome a perceived weaknesses in qualitative differential equation (QDE) based reasoning. It is designed to allow reasoning about a system despite incomplete knowledge. This is possible by establishing the influence each variable has on another, through a set of influence weights. In order to allow for the uncertain nature of a relationship in a complex dynamic system, where the system is only partially understood, fuzzy logic is used. The principle drawback of this approach is that it requires a causal network to be provided by an expert.

KHEOPS [143] A RBES. It compiles at the control level in order to achieve a reduction in complexity, a perceived weakness of practical RBESs. In order for this to be possible, the knowledge representation must be formalised and limited. Inputs are discrete or continuous, but must be within a set range, while time is strictly discrete. This keeps the complexity of the problem constant, and makes it possible to ensure that the time taken to map inputs to outputs is kept below the sampling period. This allows consistent and timely processing of input data.

IxTeT [144][145] A pattern recognition system. It is capable of recognising previously seen patterns in data, and records them in a chronicle. Knowledge is represented as logical rules. It possesses a "chronicle recognition system" which uses chronicle models, derived from the temporal characteristics of successive similar events, to allow it to predict future events of similar type.

5.2.2 PlantProtech

PlantProtech is a vibration monitoring platform for rotating machinery developed by Beran Instruments [128]. Several hardware and software elements combine to constitute a deployed platform. Hardware consists of a 76x series analyser, with varying specifications depending on the application, but all offering multi-channel vibration analysis.

Software elements include PlantProtech Vision, which is capable of visualising data and trending over time (figure 5.4). The data can be visualised as an orbit plot, or transformed into the frequency domain using the fast Fourier transform (FFT) and visualised as a Bode or polar plot.

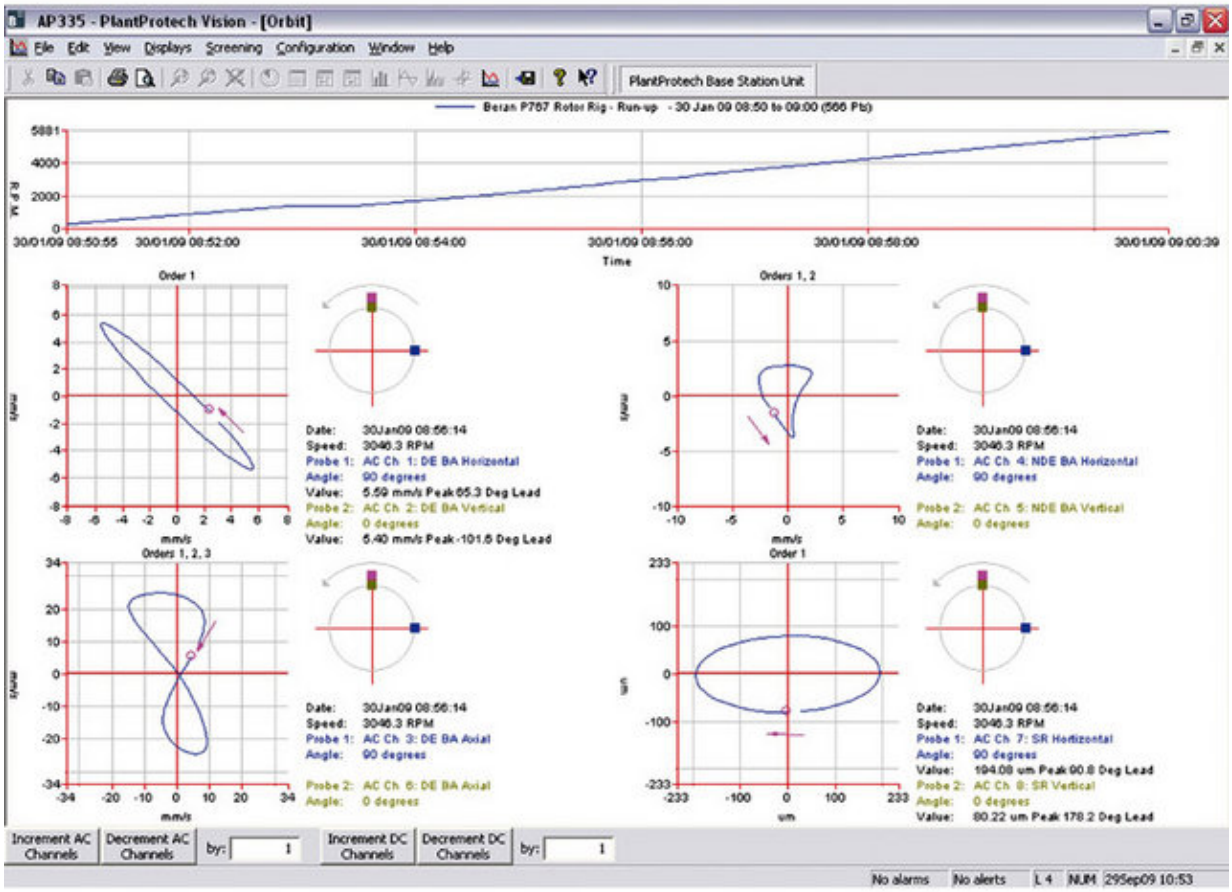

Figure 5.4: PlantProtech Vision showing trending information with several FFTs [128].

Also available is PlantProtech Mimic, a software tool designed to provide a graphical representation of the rotating machinery annotated with several key variables (figure 5.5.

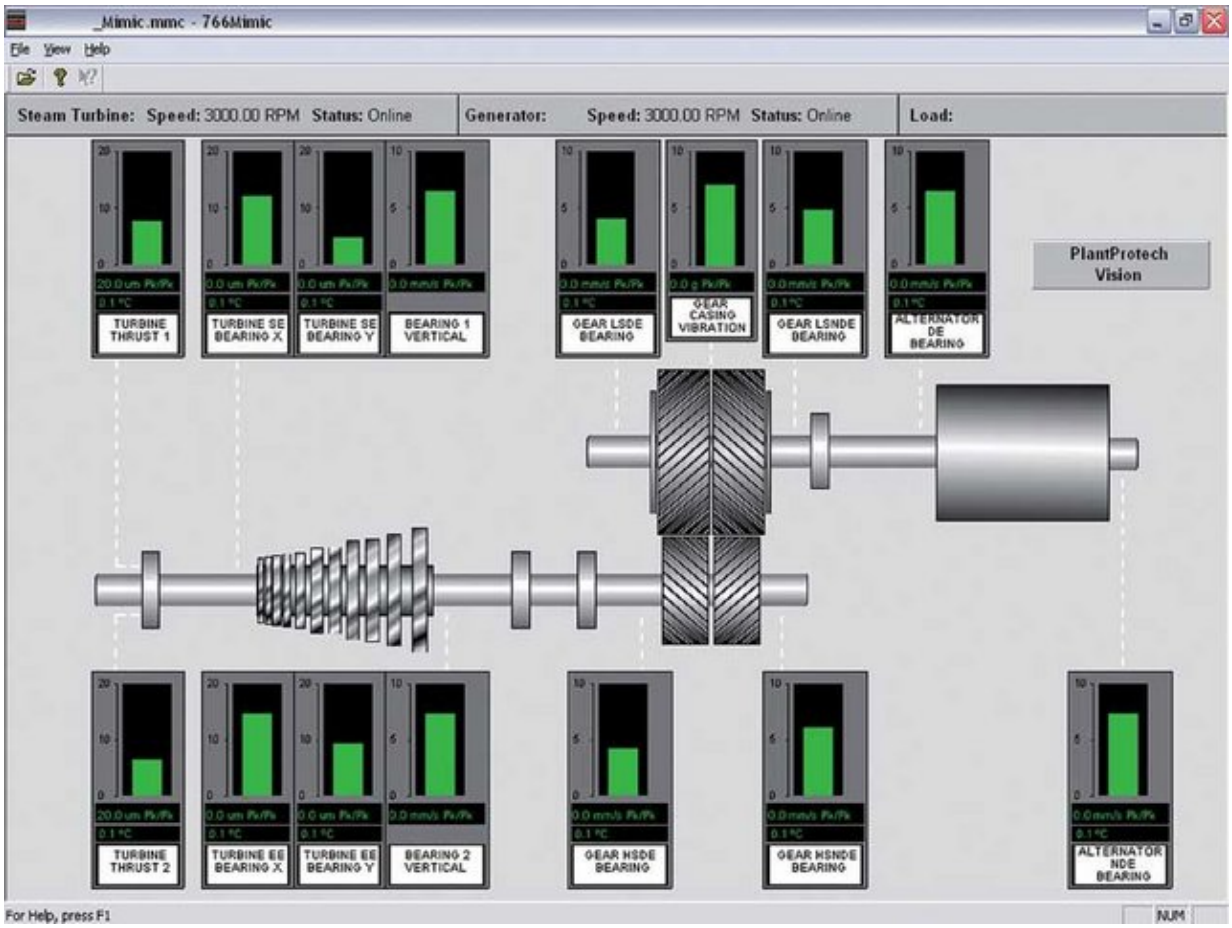

Figure 5.5: PlantProtech Mimic showing a visualisation of a steam turbine and generator connected by a parallel shaft gearbox [128].

Thanks to the use of continuous on-line vibration monitoring, changes in the health of the plant are likely to be detected early. However, the techniques used rely on high frequency data, with considerable emphasis based on FFT plots and Bode amplitude and phase plots. For example, Vision allows FFT plots of up to 500Hz. If we take this to be the Nyquist Frequency, then it would require one sample every millisecond to fully represent the spectrum [127]. As the data available for the system proposed in this thesis is sampled at most every 60 seconds, this type of monitoring is not applicable.

5.3 Plant and Data Description

The application is the gas turbines of a single unit of a CCGT plant operated by SSE. This contained two General Electric MS9001FA gas turbines rated at 228MW each, burning natural gas, each coupled via a shaft to its own generator. The exhaust gases from each turbine also feed separate heat recovery steam generators (HRSGs). The steam produced is then used to power a single steam turbine and its coupled generator.

Time-series data from each plant item is stored in a plant database, which is utilised by a SmartSignal [146] condition monitoring system utilising similarity-based modelling (SBM) [130]. The existing SBM-based system uses single timestep measurements, and is not able to model the dynamics of the GT. The proposed system will use this same data to model the dynamics of the operational turbines. The sampling rate of the data may be changed, but typical rates are between one and ten minutes. The data being used here was sampled at 10 minute intervals. While frequency-based vibration monitoring of the GTs is performed, this was not available for use with the proposed system.

The gas turbines are divided into five variable groups for modelling by the current system. All include the GT load and ambient temperature at the plant. More specifically, the variable groups are:

1. Compressor: 19 variables consisting of mostly pressures and temperatures around the axial-flow compressor.

2. Combustion: 46 variables consisting of a combustion chamber temperature vector of 31 readings, and additional input and output temperatures and pressures.

3. Fuel Flow: 21 variables including servo feedback readings, fuel temperature and pressures, and some GT operational parameters.

4. Mechanical: 30 variables primarily concerning ball bearing temperature and vibration measurements.

5. Wheelspace: 21 variables, all thermocouple readings.

All of the algorithms presented focus on a particular subset relevant to their particular problem. Each section also utilises different time periods of data.

5.4 Anomaly Detection of Exhaust Gas Temperatures

The following describes the process of analysing the data and developing an appropriate anomaly detection algorithm for exhaust gas variables, using low-frequency data.

5.4.1 Clustering and Visualisation

Following initial familiarisation with the data, An attempt was made to use SOM to identify a clear visual distinction between normal and abnormal behaviour. The SOM was trained on the exhaust gas variables from Table 5.1. The data comprised a total of 10 months of readings, sampled once every minute.

Table 5.1: Exhaust Gas Variable Set

Variable Description	**Units**
Gas Fuel Flow	Kg/s
Compressor Pressure Ratio	Ratio
Absolute compressor discharge pressure	Bar
Combustion Reference Temperature	C
Combustion Monitor Actual Spread 1	C
Combustion Monitor Actual Spread 2	C
Combustion Monitor Actual Spread 3	C
Ex Temp Median Corrected by Average	C

GT1 data from month one was utilised. This was because it contained both scheduled and tripped outages, including a trip near the end explicitly related to exhaust gas temperature. It was hoped that faults would display differing behaviour to both normal on-line data and that from scheduled outage behaviour, which could be attributed to one or more clusters generated by the SOM.

The U-matrix, shown top left in Figure 5.6, suggested two clusters; however, these corresponded simply to "off" and "on" states, with no clear anomalous cluster. The relationship between variables could not be easily visualised using a technique such as this. This suggested a complex and subtle relationship between variables. The relationships may also be linear or non-linear.

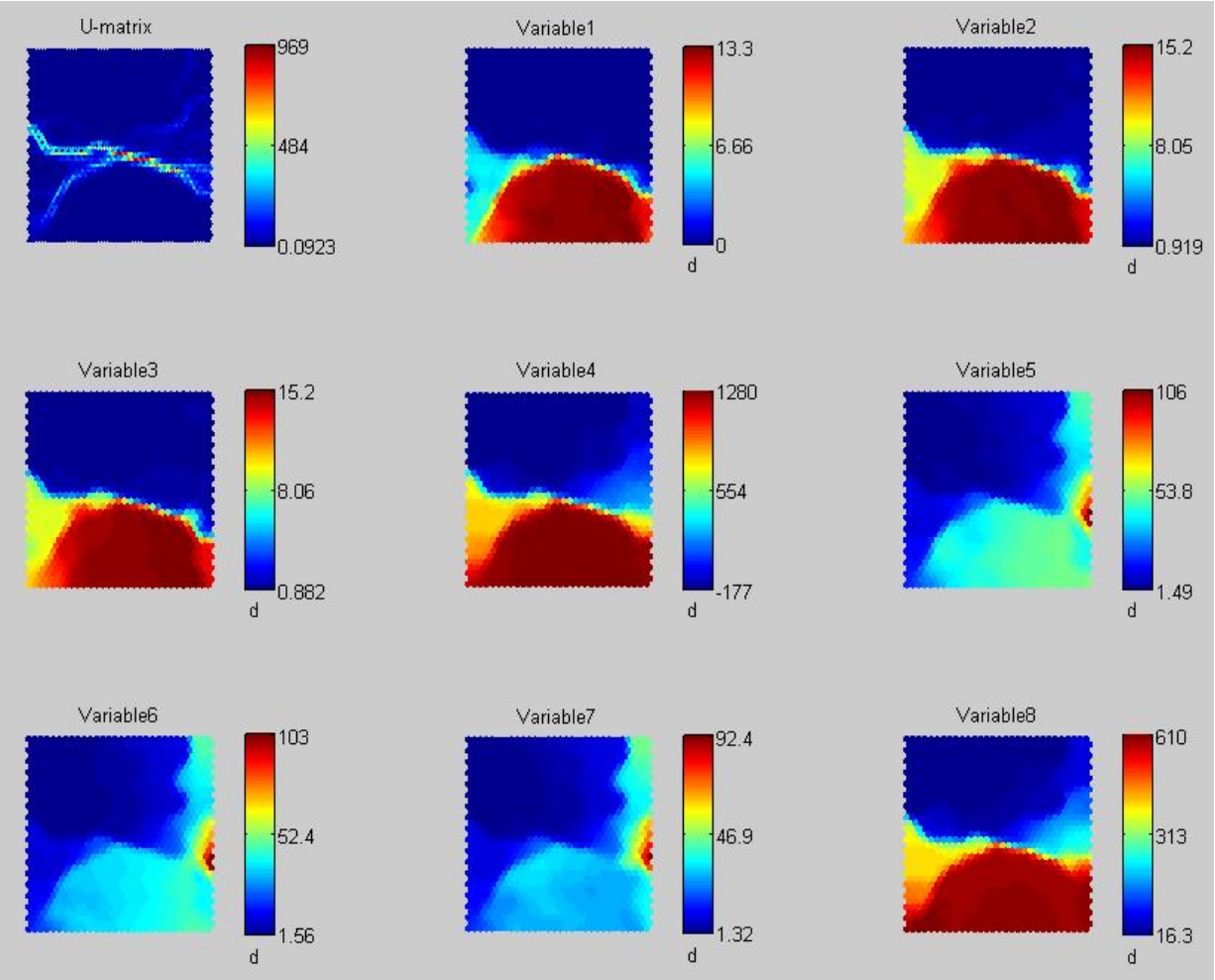

Figure 5.6: Self Organising Map for GT1 Exhaust Gas Variables.

5.4.2 Anomaly Detection using ANN-Generated Residuals

The results of visualisation of the exhaust variables suggested a complex and potentially non-linear relationship. Based on an abundance of data, compared to a lack of models and the time-consuming nature of developing an equivalent KBS, it was prudent to use a machine learning algorithm. Artificial Neural Networks were again chosen, following their successful application to pulverising mills (chapter 4).

It was attempted to use a feed-forward ANN to learn the relationships between exhaust temperature variables under normal operating behaviour, with the aim that a breakdown of the relationship could be identified before a failure. The exhaust temperature median (corrected by average) was used as the classifier. The individual residual for each time step was directly output, rather than a correlation coefficient or other met-

ric of similarity (mean square error etc.). This was as a result of requests by the utility, which requested residuals so that results could be directly compared with other methods they have used.

The data was split in two, with months 1-5 used for training data, and months 6-10 for testing data, using the exhaust gas variables in Table 5.1. As GT2 was suspected to have a pre-existing fault, all training data was taken from GT1. While this potentially means that GT2 will generally have higher residuals, if the ANN generalises properly, then a clear distinction between normal and abnormal behaviour should still be evident when testing on GT2. Full load data and normal start-up and shut-down data was used to train the ANN.

Testing consisted of two distinct phases to identify the make-up of the input variables to the ANN, and to evaluate its anomaly detection capability respectively.

5.4.3 Testing using Delayed Time Variables

The first stage of testing was the process of identifying the correct number of time delayed variables to be utilised. The use of previous values was intended to provide the ANN with additional information on how the signals change over time. It was expected that accuracy would improve given additional information.

The test was run three times, each time additional time-delayed values were added for every input variable. The first run used only the current time sample, labelled as "t". The second set used the immediately previous samples for every variable in addition to the current time sample, and was labelled "t-1". The final data set incorporated the t, t-1 and t-2 samples for all the input variables, and was labelled "t-2".

5.4.3 a Results

The ANN was tested on a scheduled outage taken from the test period. The deviation should be small, as this is not considered an anomaly.

Figure 5.7 shows a scheduled shut-down, to which residuals should be as low as possible. The load is overlaid in purple to better illustrate the behaviour of the system output.

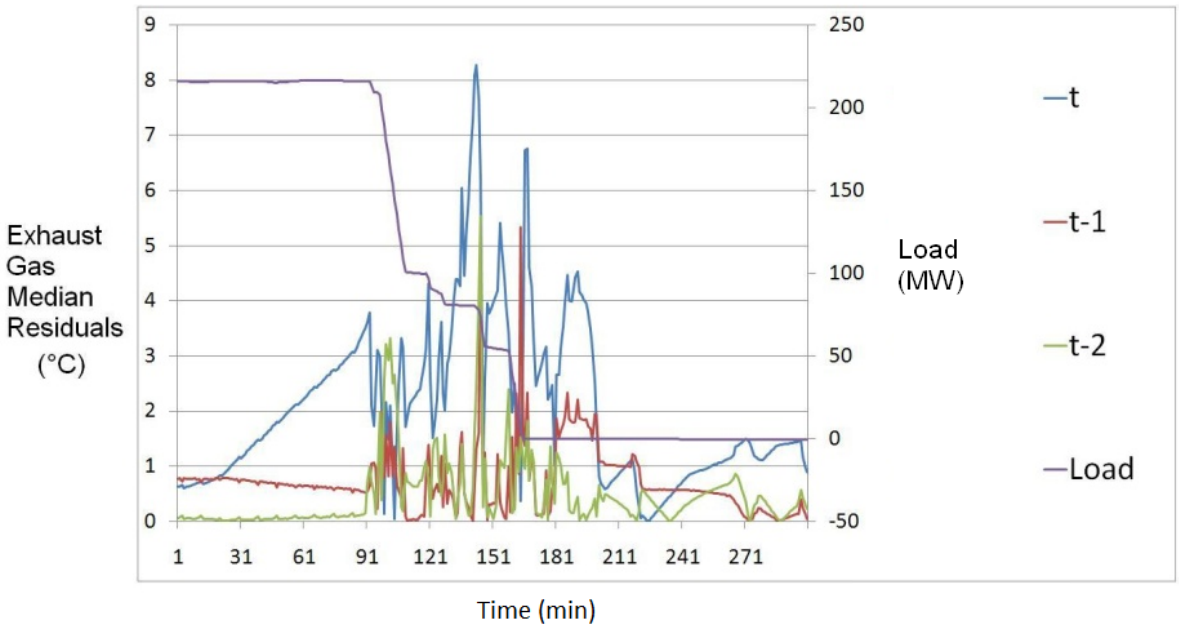

Figure 5.7: Comparison of use of delayed values for ANN input. Data from a GT1 scheduled shut-down. Load trace shown in purple, using right axis in Megawatts.

5.4.3 b Evaluation

The t-1 and t-2 residuals are lower, suggesting better accuracy. Greater delays were experimented with, but increased the training time while not improving accuracy proportionally. For this reason, a t-1 level delay was considered the best compromise and is implemented in subsequent test cases.

5.4.4 Anomaly Detection Capability

The second testing phase is testing the trained ANN on a number of trips from the test period to evaluate its performance as an anomaly detection technique. Any failure should cause a large residual between ANN output and the actual value, as this abnormal behaviour is not reflected in the training set. Conversely, normal shut-downs should minimise the residual.

5.4.4 a Results

The residuals for testing, using failures which required unscheduled maintenance or repairs in month six and ten, are shown in Figures 5.8 and 5.9 respectively.

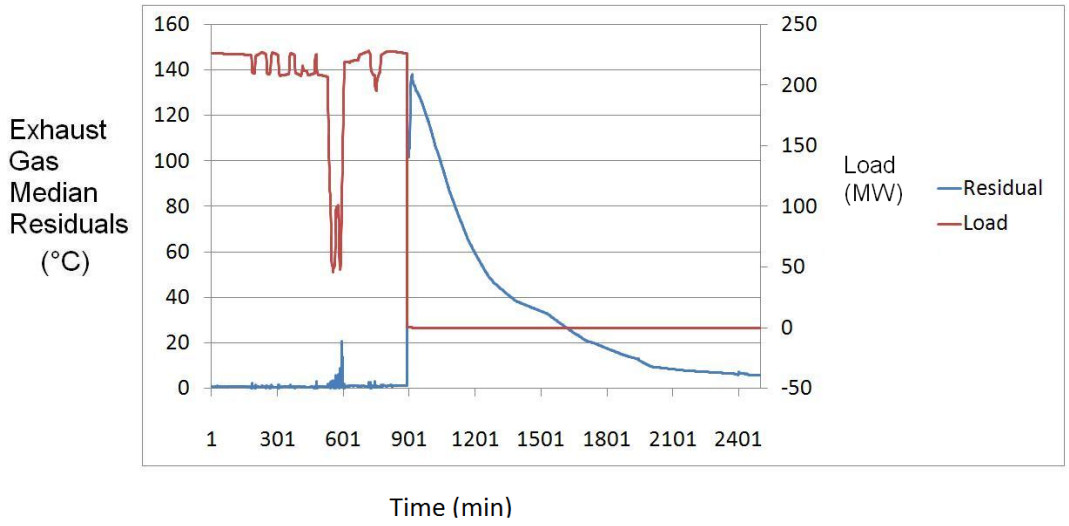

Figure 5.8: Residuals for GT2 in month 6. Residual spike due to trip for high exhaust temperature and vibration.

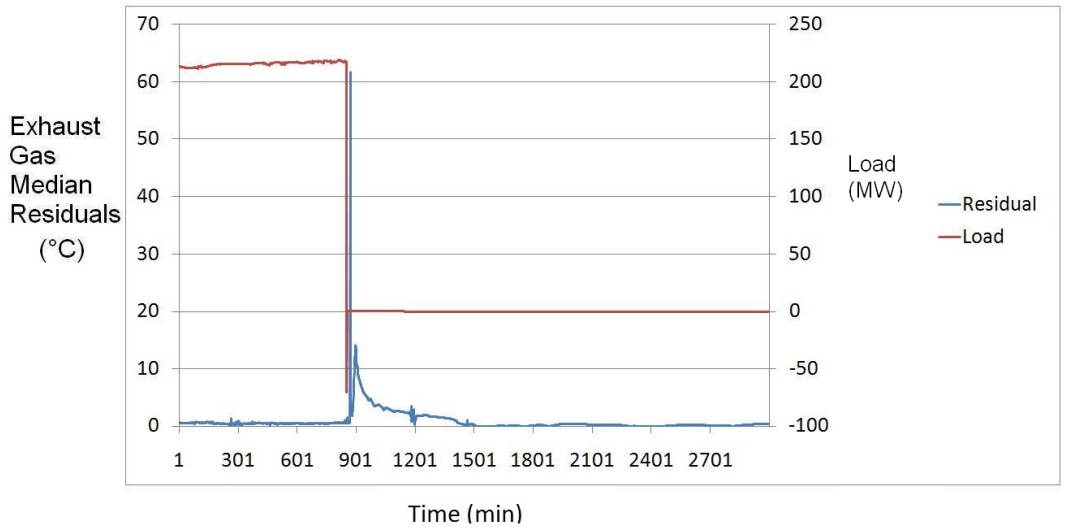

Figure 5.9: Residuals from GT2 for month 10. Spike corresponds to trip related to exhaust gas temperature.

Figure 5.10 shows a scheduled outage. Figure 5.11 shows detection of an exhaust gas anomaly. Figure 5.12 shows what was thought to be a false positive, but upon examination with a load trace can be attributed to a sensor error shown in the negative load values. Figure 5.13 shows a trip not related to exhaust gas temperature.

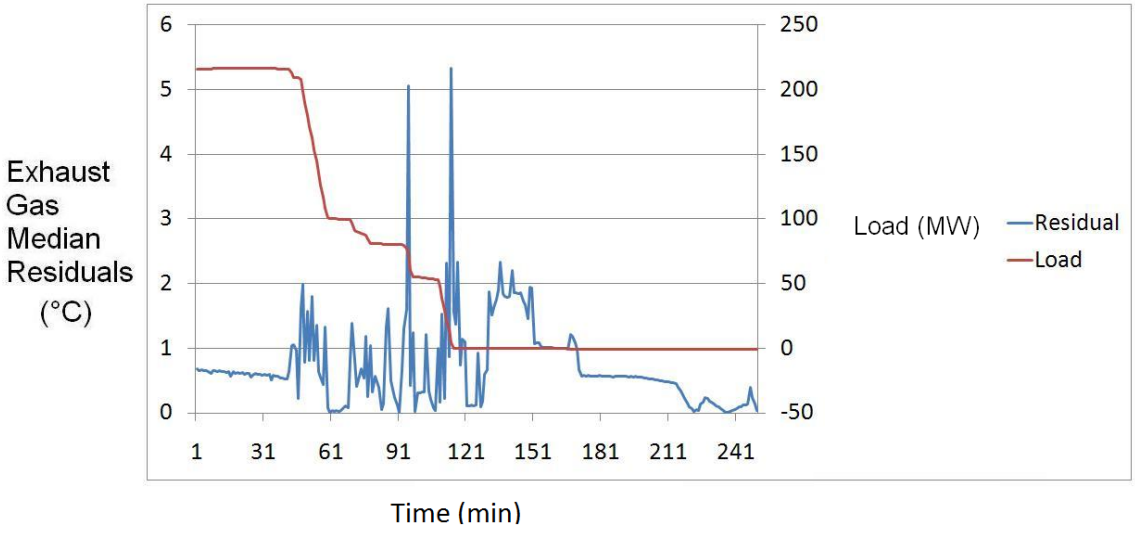

Figure 5.10: Scheduled Outage from GT1. Note the small value of the residuals throughout.

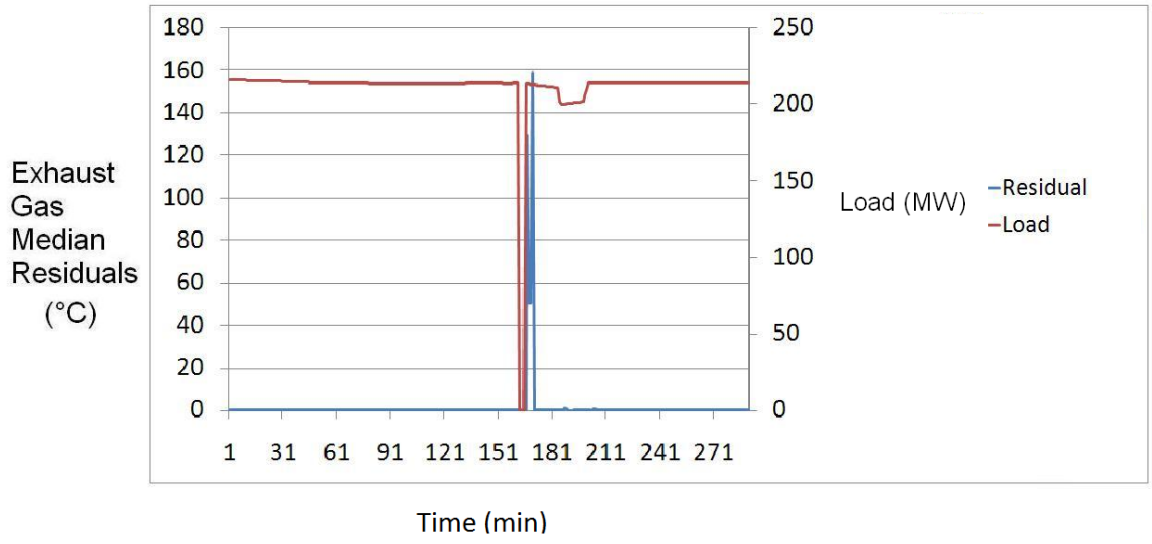

Figure 5.11: Residual spike following trip related to GT1 exhaust gas temperature.

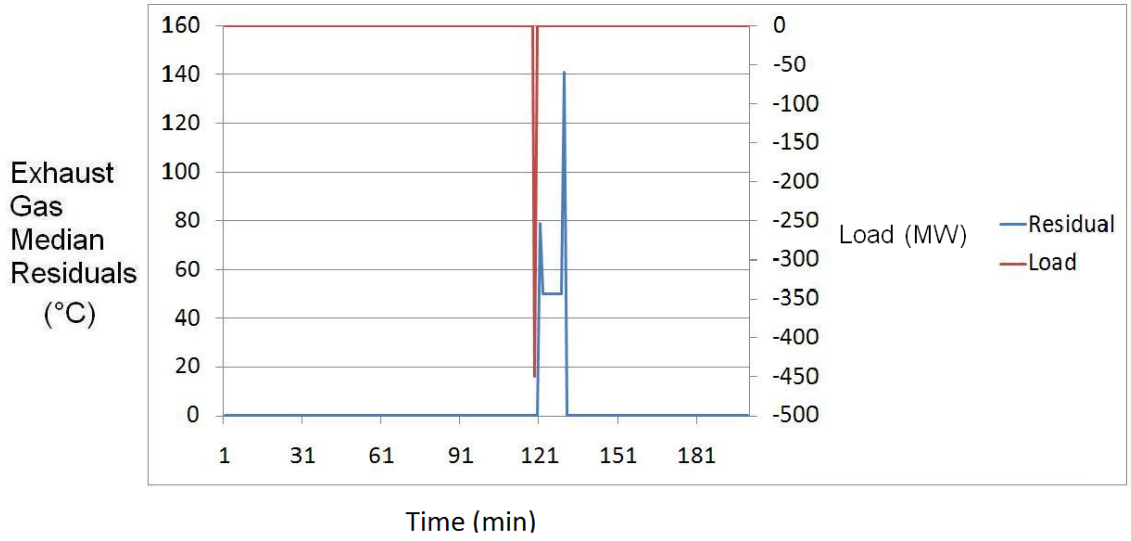

Figure 5.12: A residual spike; caused by an error in the GT1 load value resulting in high negative values.

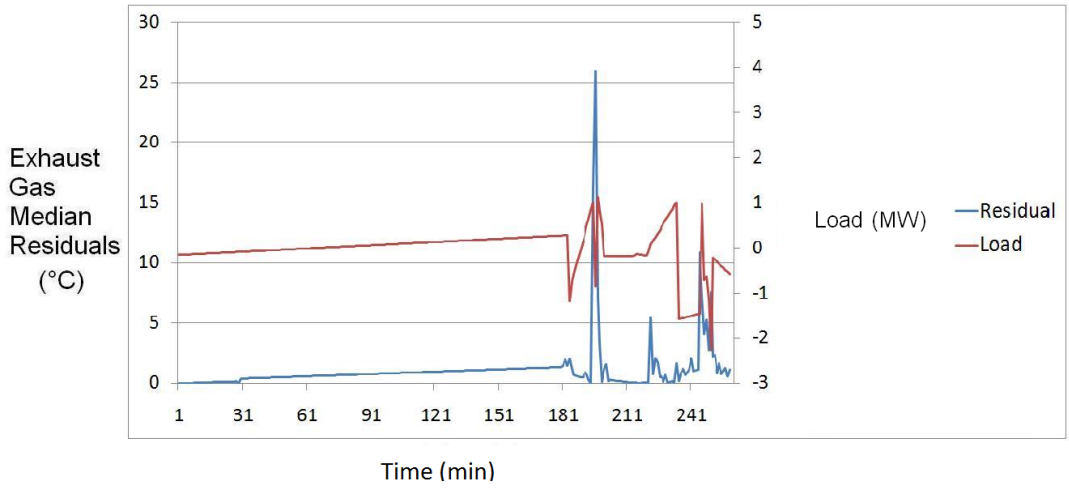

Figure 5.13: A GT1 trip unrelated to exhaust gas, with very high residual.

5.4.4 b Evaluation

The results show good identification of anomalies within five minutes of occurrence based on visual identification. The technique could be used to automatically raise alarms, based on the residual exceeding a certain threshold. Typical values of normal operation residuals are usually less than 10, while the largest spikes can be in excess of 100. This represents a good dynamic range, which should make it easy to identify when

to raise an alarm. Based on the results from testing carried out on the entire 10 month period for GT1, a threshold of 10 is suggested.

Figures 5.8 and 5.9 show a clear "spike" in residuals during a known anomaly. The fact that the training data was from GT1, while both failures were in GT2, suggests good generalisation of the ANN. These faults previously went undetected before failure, and demonstrates the technique's capabilities

Figure 5.10 shows a normal shut-down, where the residuals are never greater than the proposed automated threshold of 10. For comparison, Figures 5.11 to 5.13 show a number of other faults, all of which break this threshold, suggesting that the automated algorithm is capable of distinguishing between normal and abnormal shut-downs. The ability to detect an anomaly unrelated to exhaust gas (Figure 5.13) suggests that the technique may be applicable to other kinds of faults.

5.4.5 Analysis of Performance

The ANN has shown to be capable of successful detection of exhaust gas anomalies, where a current system was unable to do so. The proposed algorithm has done so, despite the absence of high frequency data.

5.5 Modelling a Compressor using HMMs

The second proposed condition monitoring algorithm for gas turbines is aimed at modelling the compressor, in order to detect faults and monitor degradation of the machine.

5.5.1 Data Familiarisation, Visualisation and Reduction

The compressor variables from Section 5.3 were used as a starting point. A new set of data was provided, and once again comprised 10 months of data, this time sampled every 10 minutes. Analysis and visualisation of the variables was carried out in order to improve understanding of the data. It became clear that many of the variables were closely linked, both in terms of position or purpose in the turbine, and statistically. Following the advice of a gas turbine expert from the utility, the available 19 variables were reduced to those shown in Table 5.2.

In addition to this, on the advice of the GT expert, two additional variables were made available, and are shown in Table 5.3.

Table 5.2: Compressor Variables

Variable No	Description	Units
1	Ambient Temperature	^oC
2	Ambient Pressure	bar
3	Load	MW
4	Compressor Inlet Temperature	^oC
5	Fuel Flow	kg/s
6	Absolute compressor discharge pressure	bar
7	Compressor Discharge Temperature	^oC

Table 5.3: Additional Compressor Variables

Variable No	Description	Units
8	Inlet Guide Vane	^oC
9	Heat Valve	bar

To determine the impact these additional variables may have on algorithm performance the two sets (Original (1-7) and Extended (1-9)) are compared in subsequent tests.

Exclusively on-load data was used for analysis. This was motivated primarily by the abundance of such data, due to the long periods of time a GT spends on-load. This would provide ample training data, a pre-requisite when using a data-based technique[147]. Run-up and Run-down data was considered, but this was less common and more intermittent. Attempting to visualize this data showed that the scarcity and brevity of these occurrences would make the trending of GT behaviour, and therefore the detection of degradation in performance, more difficult.

5.5.2 Algorithm Selection

A probabilistic model was chosen for its ability to model the GT and its underlying behaviour. This eventually resulted in HMMs being chosen for evaluation, driven by their ability to model this underlying behaviour and dynamics through their use of hidden states. It was expected that this would allow them to outperform other techniques in timely anomaly detection. While ANNs have shown fault detection capability, they have shown little capability to detect precursors to pending faults [15] greater than a few minutes.

5.5.2 a Designing the HMM

Following the selection of HMMs, several design decisions had to be made. Firstly, the type of HMM had to be picked. Discrete HMMs were chosen at this point due to the relative simplicity compared to other models, such as those utilising Gaussian distributions and/or multivariate inputs. More advanced models were considered, but for the first pass of testing, it was decided that the principles of the HMM be proven using the simplest model possible.

With a discrete HMM, a method of discretising the data has to be chosen. K-means clustering was chosen due to the well known properties of this method. A summary of the process of generating the observations is provided in section 5.5.2 b By using a well-understood clustering method, the behaviour of the HMM would not be obscured behind less familiar, but more advanced, methods.

The data was sampled every 10 minutes, as provided by the utility. For SmartSignal, the data is grouped into three hour blocks. In line with this the HMM grouped the samples into observation sequences of length 20, resulting in a period of three hours and twenty minutes. The HMM was set to eight states, and ten possible discrete observations.

5.5.2 b From Clusters to Observations

In order to create observation sequences from raw data, it is necessary to cluster them. This was done with K-means clustering. The training data is used to define a number of clusters, and then each test data point is classified into a cluster (Figure 5.14).

A number of contiguous samples are taken and their cluster values form a new observation sequence, as shown in Figure 5.15. An observation sequence, O is therefore a vector of integers. The values correspond to the nearest K-means cluster to the original data sample. The order of the numbers are of no particular significance, as they are just cluster labels, and could in fact just as easily be characters or symbols such as “A”, “B” and “C”.

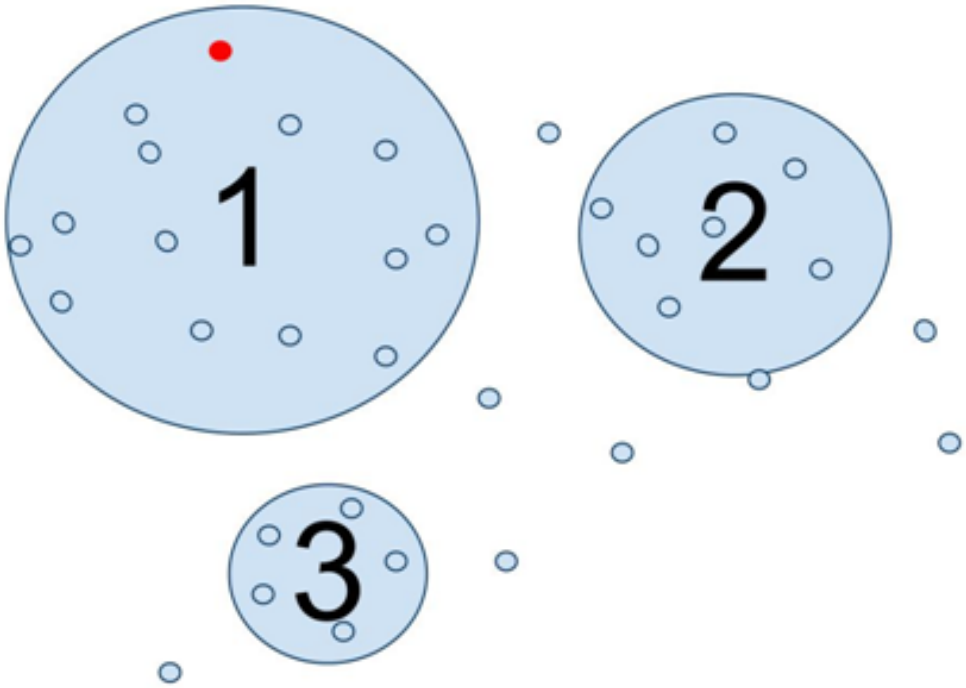

Figure 5.14: Illustration of clusters. The red point is classified accordingly as Cluster 1.

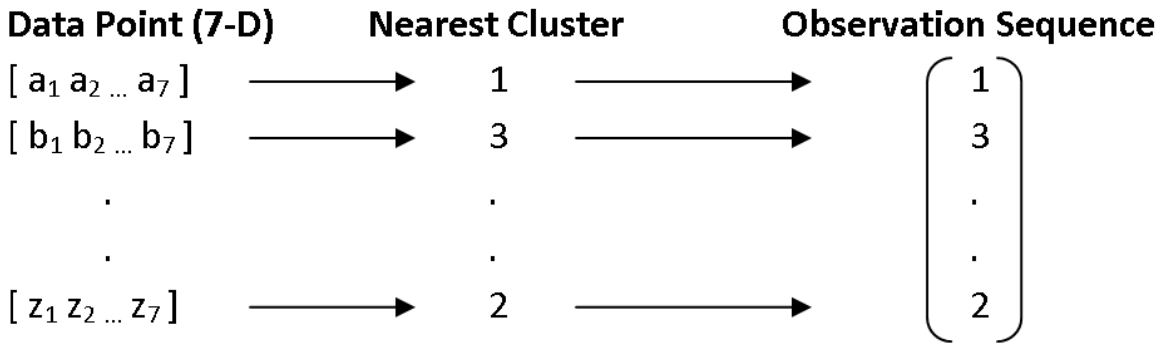

Figure 5.15: The process of classifying a data point and entering into an observation sequence

As an HMM has no direct mapping between an observation and a state, the Baum-Welch algorithm [53] as described in Section 2.8.2 b is applied. It is used to determine probabilities for both starting in and transitioning between states, allowing us to define A and π, and the probabilities of seeing particular observations for those states, deriving

B.

In the case of a discrete HMM derived from clusters, each row of the matrix B represents a state, with the columns corresponding to a cluster. $b_j(O)$ is therefore equivalent to the probability an input falling into cluster "O" when in state "j".

With A, B and π defined, the model λ is completely specified (Equation 2.4) We now have a probabilistic model of the underlying behaviour of the training data.

During testing the forward term α (Equation 2.8) is used to calculate the probabilities of each observation sequence. The data points are broken up into contiguous blocks of 20, before each point is clustered, providing $O_1, O_2...O_{20}$. The corresponding forward-term values $alpha_1, alpha_2...alpha_{20}$ are used as the output of the model.

5.5.3 Testing the Compressor Model

The following tests were carried out, using the Matlab HMM Toolbox [148], to determine the capability of HMMs to detect faults and machine degradation.

5.5.3 a Methodology

The available data was split into three sub-sets (A, B and C), divided by time. Subset A was chosen for training as this represented a period when the turbine was believed to be operating under normal operating conditions, with no significant issues. The remaining two subsets were used for testing. When visualising the results the log-likelihood was plotted for each sequence, in order to identify a "cone" of normal behaviour[5]. It was expected to see low LL values, below this cone of normality, for anomalous behaviour.

5.5.3 b Results

The results for both GTs across the three subsets are shown in Figures 5.16 and 5.17.

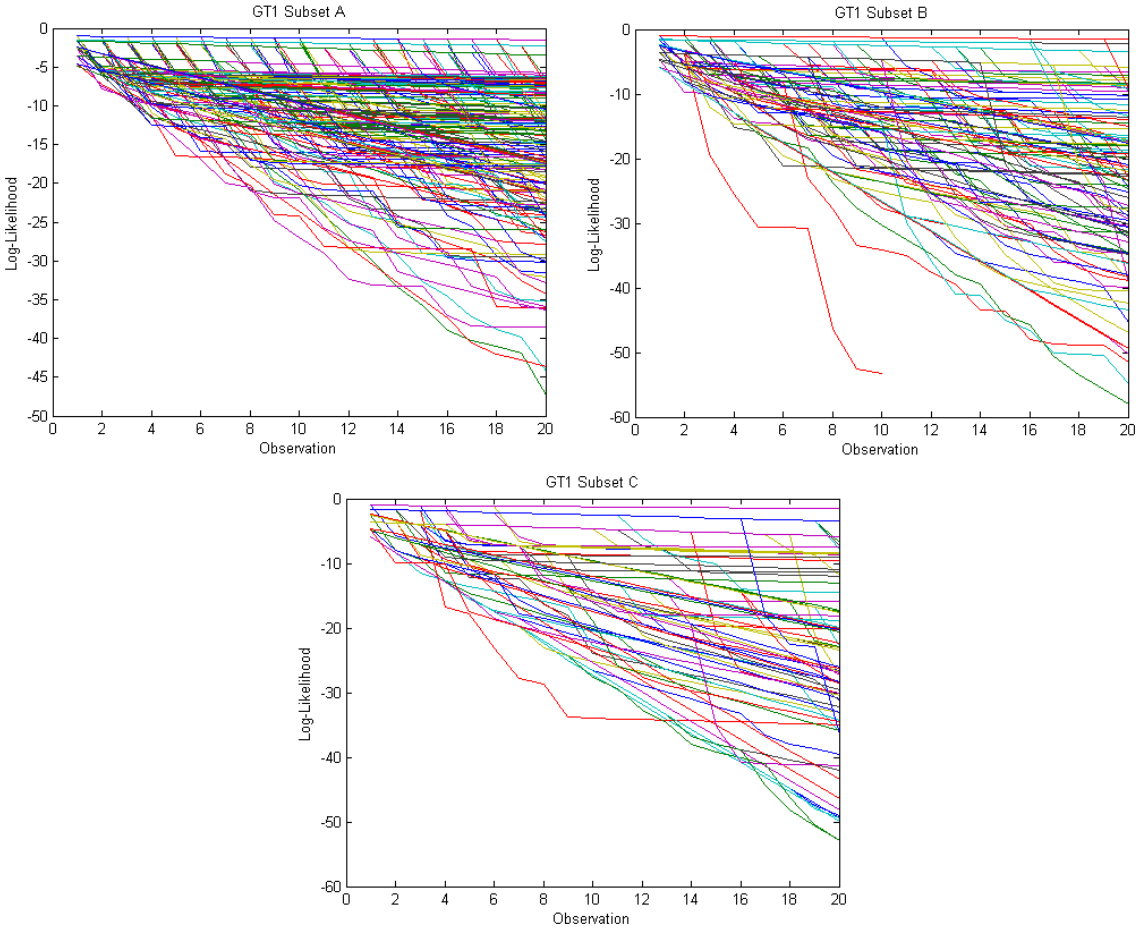

Figure 5.16: GT1 DHMM test results. Note the LL is above -60 in all cases.

5.5.3 c Evaluation

Figure 5.16 shows GT1 maintains a LL of greater than -60 throughout the test period. Considering the period of study was over several months, this shows impressive resilience to seasonal changes.

It was known that GT2 was suffering performance degradation believed to be related to compressor fouling towards the end of the available time period. This results in sequences deviating from the cone of normality. In fact subsets B and C in Figure 5.17 show several sequences that are much lower in LL than subset A. The Figures suggest two groups of values: those above -60, and those below. At first this was suspected to be due to a seasonal change, as the subsets span several months each. However, there is no similar behaviour reflected in GT1, suggesting this lower cone is representative of a fundamental change in GT2's behaviour away from the training examples.

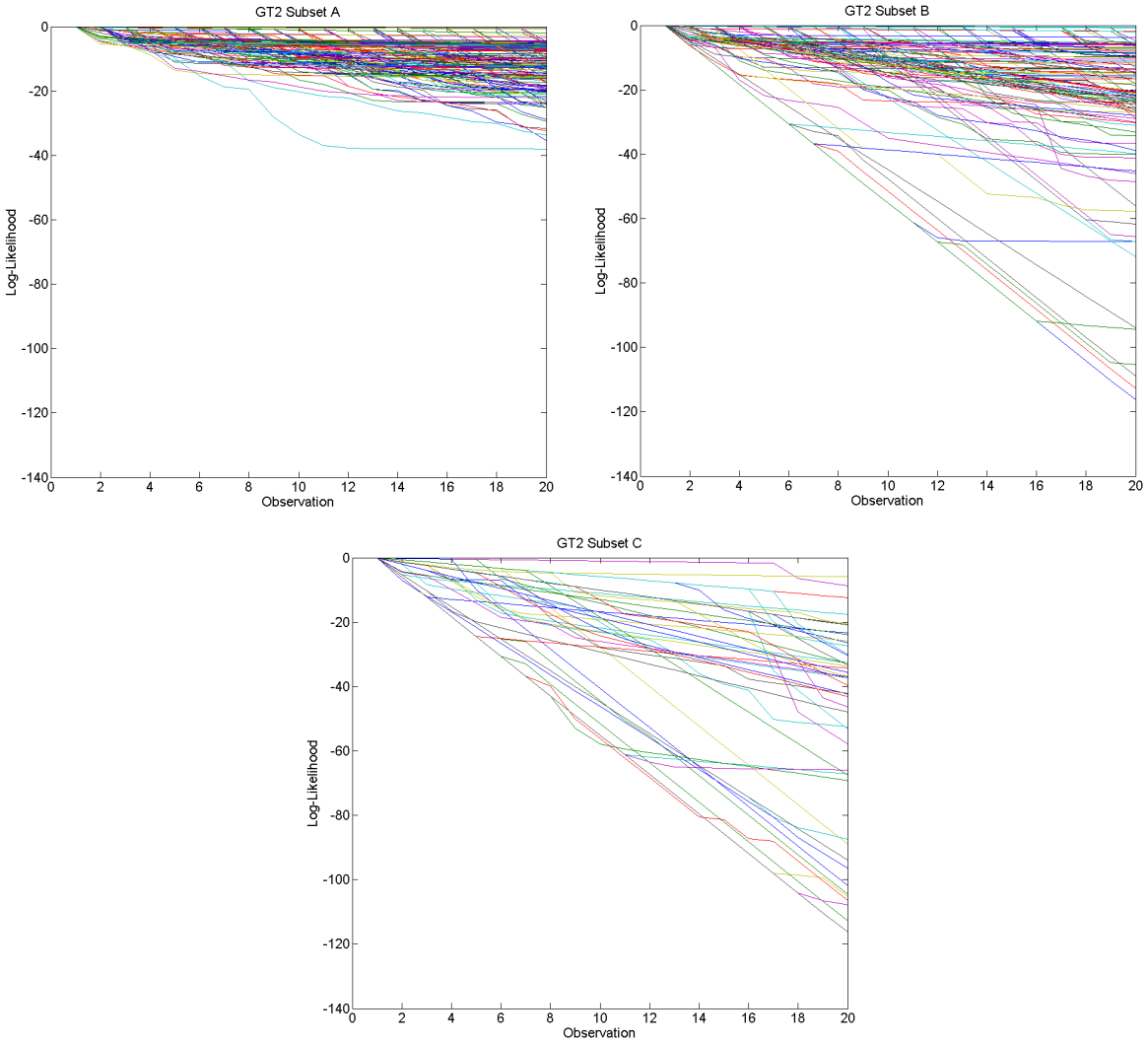

Figure 5.17: GT2 DHMM test results. Note the low LL in Subset B and C

5.5.4 Automation of Anomaly Detection

The results from GT1 and GT2A in Figures 5.16 and 5.17 would suggest that a normal sequence ends with a LL of no lower than -60. From studying the results, it seemed that most sequences fell into the cone of normality or fell to a final LL (observation 20 of each sequence) of around -100 to -120 with very few examples of values in between. With this in mind, the final LL of GT2 was thresholded using -60. A visualisation is shown in Figure 5.18.

The results show both the original and extended variable sets. Both show two periods of periodic anomalous sequences. After discussion with the utility, it was revealed

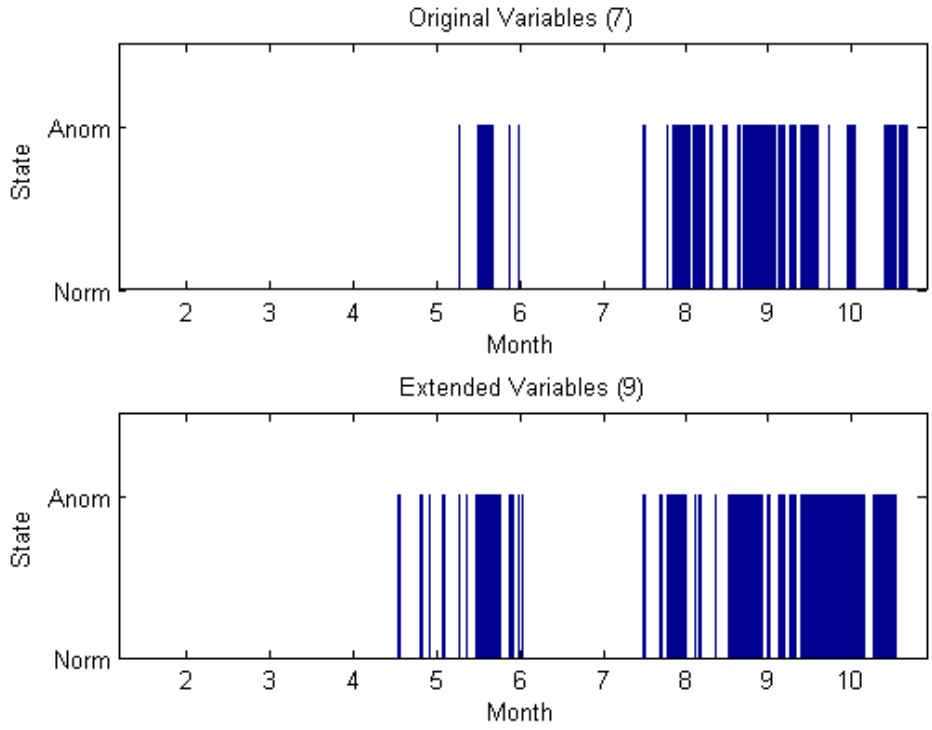

Figure 5.18: Visualisation of detected anomalies in GT2 using a DHMM

that both periods correlated to increasing levels of compressor fouling. The period between is due to down time. The extended set seems to show detection of this deterioration earlier, around the middle of month four rather than month five. This suggests that the two additional variables have improved model performance.

This thresholding approach provides a useful indication of abnormal system operation, without the need for manual analysis. This fulfils the requirement for anomaly detection, but does not provide information of machine degradation.

5.5.5 Comparison between discrete and continuous HMMs

Due to the desire to provide a clearer indicator of gradual degradation, the use of a non-discrete HMM was considered. Following is a comparison of a DHMM to a MHMM, using the compressor data.

5.5.5 a Methodology

Using the same data, a HMM using Gaussian Mixture Models for the observation probability functions was constructed. The model used the raw data as input, rather than passing through a K-means clusterer as before. There were eight states with ten mixtures per state to parallel the eight states and ten clusters used in the discrete models.

The same process as before was followed, with the important exception that rather than defining the observation matrix B, a MHMM requires us to calculate the values for mixture weight matrix C, the mean matrix M, and the covariance matrix Σ, allowing us to define a GMM for each state. The Baum-Welsh algorithm is once again applied to calculate all the necessary variables, utilising the forward-backward procedure.

The data points are broken up into the same blocks of 20 as in the DHMM, however it is not necessary to cluster them, making the observations $O_1, O_2 ... O_{20}$ a sequence of vectors. During testing $b_j(O)$, the probability of observation "O" for state "j", is calculated using Equation 2.6. With these values now calculated the forward term α (Equation 2.8) is used to calculate the probabilities of each observation sequence and used as the output of the model as for the DHMM.

5.5.5 b Results

The two HMMs are compared over the 10 month period in Figure 5.19.

5.5.5 c Evaluation

From the results shown in Figure 5.19, the use of a MHMM appears to show the degradation of the turbine over time more clearly. This is visible through the gradual downward trend in the LL values, which is not possible through the DHMM as it tended to group values into one of two ranges, corresponding to either anomalous or normal behaviour.

The low log likelihood before outages, including that at the beginning of month 6, is still present and visible in the MHMM models. There is a particularly large dip in both the original and extended variable sets in month 9. This was found to be at the time of IGV adjustment. While this event is lost in the large number of sequences with low LL values in the DHMM results (figure 5.18) around this time, it stands out clearly in the MHMM results despite the presence of a concurrent compressor fouling issue.

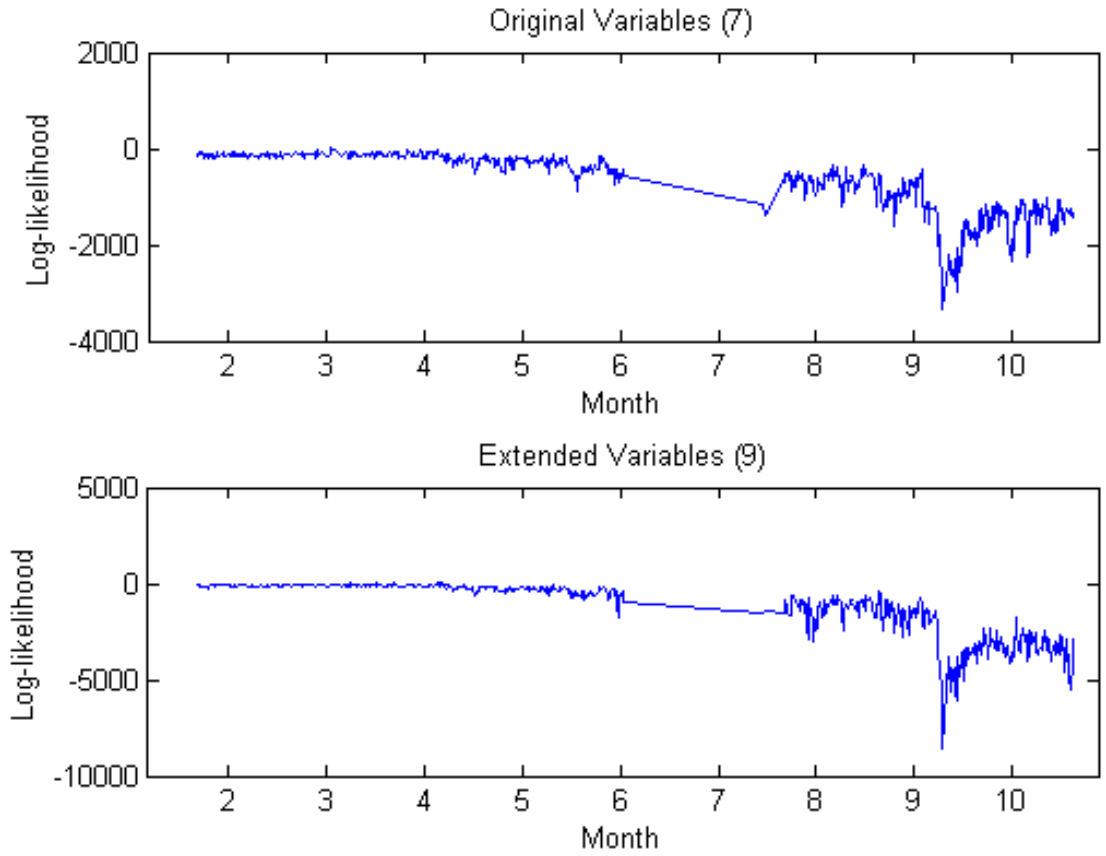

Figure 5.19: Test on GT2 compressor model using GMMs to represent observation probabilities.

The results presented in Figure 5.19 also suggest that the extended variable set is still a superior measure of condition, as normal LL values remain similar to the original, but anomalous behaviour results in lower LL. This is particularly evident during the IGV adjustment in month 9, where the original set has a LL value of -3000, compared to the lower -9000 of the extended set.

This comparison suggests that the MHMMs tend to show degradation more clearly than DHMMs by way of small changes in LL value. However, the spread of values is more varied and therefore it is hard to derive a simple automated thresholding approach such as that presented in Section 5.5.4.

5.5.6 Evaluation of Performance

The use of HMMs for modelling GT compressors has been shown to be provide useful metrics indicating machine condition. DHMMs have been shown to be able to detect anomalies using low-frequency data. However, MHMMs show greater capacity for machine degradation monitoring. Both approaches have been shown to effective condition monitoring techniques.

5.6 Modelling Combustion using MHMMs

This section presents a comparison of a HMM-based technique and the existing SBM-based system, for condition monitoring of GT combustion variables. The aim was to provide a complementary anomaly detection and degradation tracking capability in addition to the system already in place. Once was again carried out using the Matlab HMM Toolbox [148],

5.6.1 Data Preparation

Onload data was once again used for analysis, motivated primarily by the abundance of such data. A new set of data was provided, comprising nine months. The data was sampled every 10 minutes, as provided by the utility. The existing SmartSignal system used by the utility groups data into three hour blocks. In line with this the HMM grouped the samples into observation sequences of length 20, resulting in a period of 3 hours and 20 minutes.

The use of a discrete HMM was discounted as it would have required a preprocessing stage in the form of a classifier or clusterer. By its nature a classifier could reduce the sensitivity of the HMM to small changes in inputs. While this may be a positive attribute for some fault diagnosis systems, the desire for the HMMs to be able to show gradual degradation over long periods of time makes this approach poorly suited to this application. Due to the desire to provide a clearer indicator of gradual degradation a continuous HMM technique was selected for implementation and testing.

Data reduction techniques were considered, but as this would require an additional stage before the HMM that would make it more difficult to evaluate the HMM performance. It was also desirable to use the same range of inputs as the existing condition monitoring system 5.3, to provide a "fairer" comparison. Due to the presence of known

faults strongly correlated to combustion, the results presented in this section focus on the combustion variables.

5.6.1 a Processing Results

For simplicity and transparency, and the need to view a timeline to match utility requirements, only the final LL value of a sequence is considered when evaluating the performance of the algorithm. Considering each LL cone separately would be a time consuming process. However, the prototype system (Chapter 6) contains the capability for the user to isolate any point on the final LL graph and view the full LL cone if they need further information.

When evaluating the performance, it is necessary to define how to interpret the HMM results. The HMM's LL value does not have a hard threshold as part of the motivation behind using MHMMs over their discrete counterpart was their greater sensitivity reflected in the range of output LL values. Thus, not only the absolute LL value but also the general trend and any dips or divergence from that trend all had to be considered. A low absolute value could be considered a persistent fault, or could simply be a slight change in behaviour due to a maintenance procedure. The trend is useful as this can often indicate degradation which can lead up to a failure. Finally, a "dip" outside of this trend can often indicate a fault has already occurred. Considering all three of these metrics is potentially more complicated to automate than a simple threshold-based approach. In this thesis all the results have been interpreted by hand. However, several approaches to automated techniques to analyse the LL plots are discussed in section 5.8.

5.6.2 Experimentation

Nine months of data was split into three and six months of training and testing data respectively. The training data was the same as that used for the the existing SBM-based system. However, the existing system was updated with additional training samples from later periods added as part of model maintenance. As model maintenance is carried out by the system provider, this data could not be determined easily, and it was necessary to perform the comparison with the data unavailable to the HMM.

5.6.2 a Designing the Model

With the training and test data sets established, the next step is to define a satisfactory model architecture. Because of the 'black box' nature of HMMs the optimal number of states and observations/mixtures for a HMM is difficult to calculate exactly. It is also possible that the potential accuracy of a larger model is offset by the increased computational complexity that this requires. There is no hard-and-fast rule for determining the best model parameters, so it is necessary to rerun tests with different models.

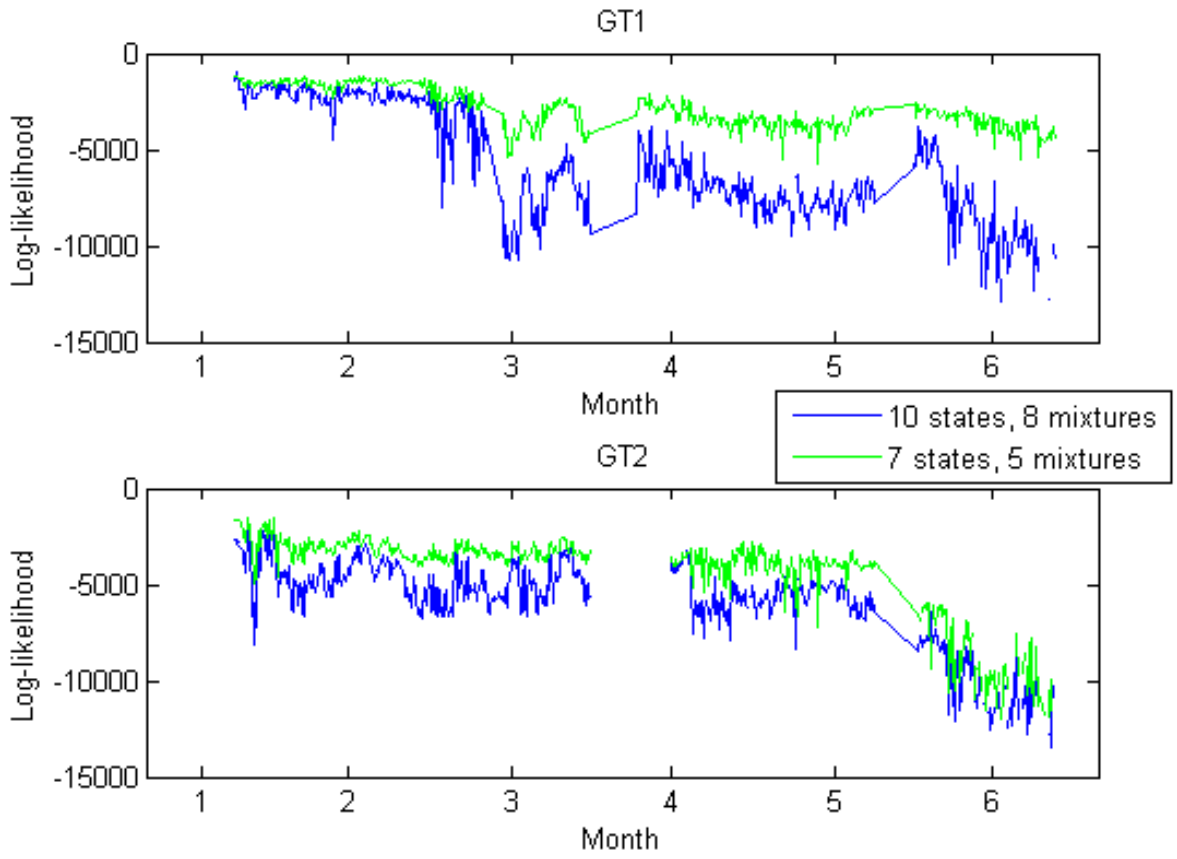

Figure 5.20: Comparison of model sizes.

The first round of tests concentrated on the size of the model and varied the number of states and mixtures. When considering both fidelity and computational expense, a good compromise was found using ten states and eight mixtures. Figure 5.20 shows that a smaller model loses some of the dynamic range between normal and abnormal LL values. While smaller models did decrease training and testing time, this was not

sufficient justification for a loss in model effectiveness.

Larger models increased the dynamic range but resulted in a large number of values dropping below that which could be represented (discussed further in Section 5.8). The greater dynamic range of large models is due to the larger number of mixtures and states allowing the model to more closely fit the training data and, in the case of mixtures, reducing the Gaussian variance. This would result in abnormal values, which would be expected to be farther from the Gaussian mean, having much lower LL values. While this is generally desirable, it also increases the probability that a single LL can drop below the range of numbers representable by the system. This was the case with some of the larger models. While it is still possible to view the sequence individually, the final LL value was the principal measure for analysis and this resulted in more of an on-off anomaly detection capability. This was not the desired outcome and for this application smaller models were deemed more appropriate.

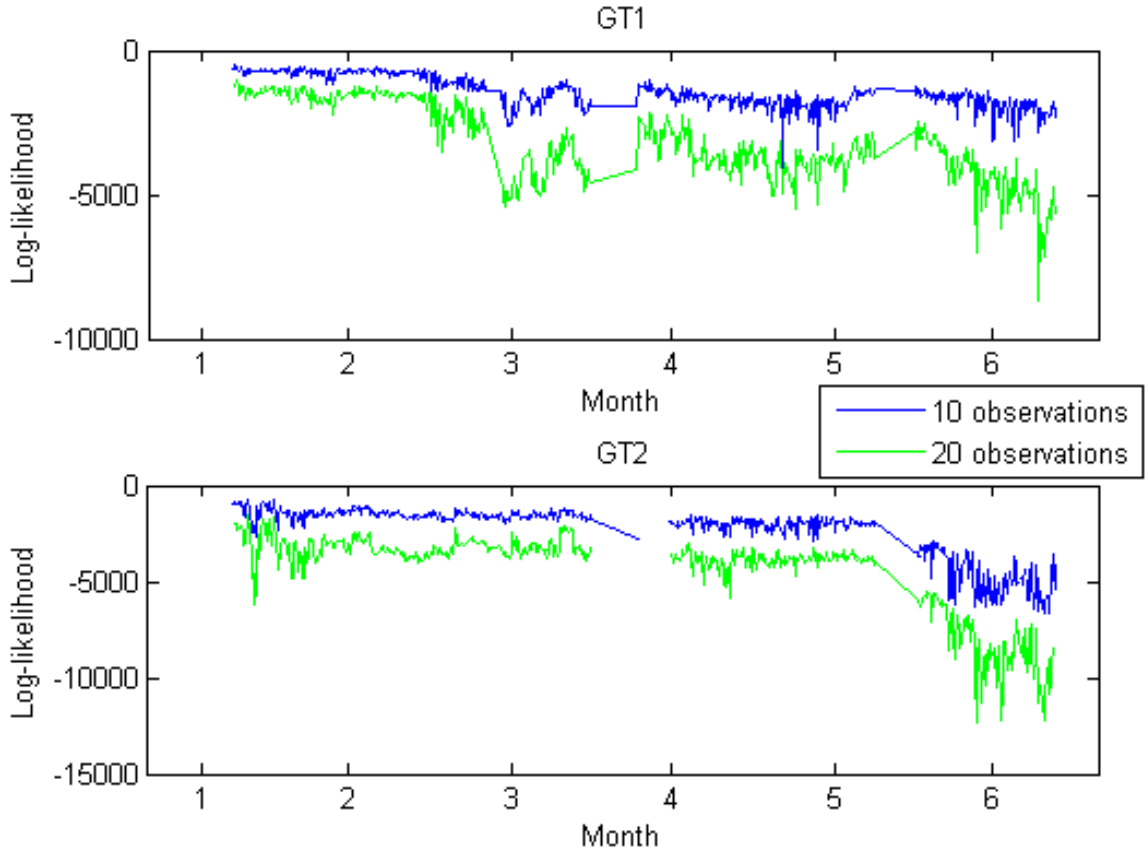

Figure 5.21: Comparison of testing with different observation sequence lengths

Further experimentation was carried out by varying the length of the observation sequences fed into the model. Figure 5.21 shows that the smaller sequence has generally higher LL values. This is because LL is cumulative. While it would also increase the frequency of model output, the difference between normal and abnormal behaviour is once again compromised when looking at the final LL values. Conversely, when tested against sequences with greater than 20 observations the number of sequences dropped and, as the LL values decreased uniformly, did not display substantially improved fidelity or dynamic range that would assist in determining the presence of anomalies. These results have led to the conclusion that the original sequence length of 20 observations is adequate. As this is also in line with the utility's practice of analysing three hours of data at a time this was used for all subsequent tests.

5.6.2 b Validation

After some preliminary tests, promising results were obtained for both turbines using models with ten states and eight mixtures, using the sequence length of 20. These results are shown in Figure 5.22.

It was attempted to correlate the level, trends, dips and variations in LL values shown in Figure 5.22 to trips and other factors that may explain them.

GT1 starts with an initially higher LL value than GT2, but drops towards the end of month two (point A). This is believed to be linked to low level faults, that would affect previous data (including training data), that were fixed during month 2. This would affect the behaviour of the turbine and would be visible as a drop in LL.

While both turbines experience an outage starting in the middle of month three for routine maintenance, GT2 appears to start later after the month three shut-down (B). In fact this is due to several trips running up to end of the month. It was rarely up to base load long enough for a complete sequence and there were very low LL values (below what could be represented, suggesting extremely unusual behaviour) when it was.

Both turbines show a slight drop in month four which correlates with combustion retuning changing the GT behaviour slightly (C). There is an outage in month five (D), which shows GT1's LL temporarily increase. This is believed to be due to maintenance performed on GT1 during this time. The values of both GT1 and GT2 then gradually drop off until increasing number of extremely low unrepresentable values begin to occur (E). This is due to a combination of low level faults and a particularly large seasonal variation in temperature and pressure values.

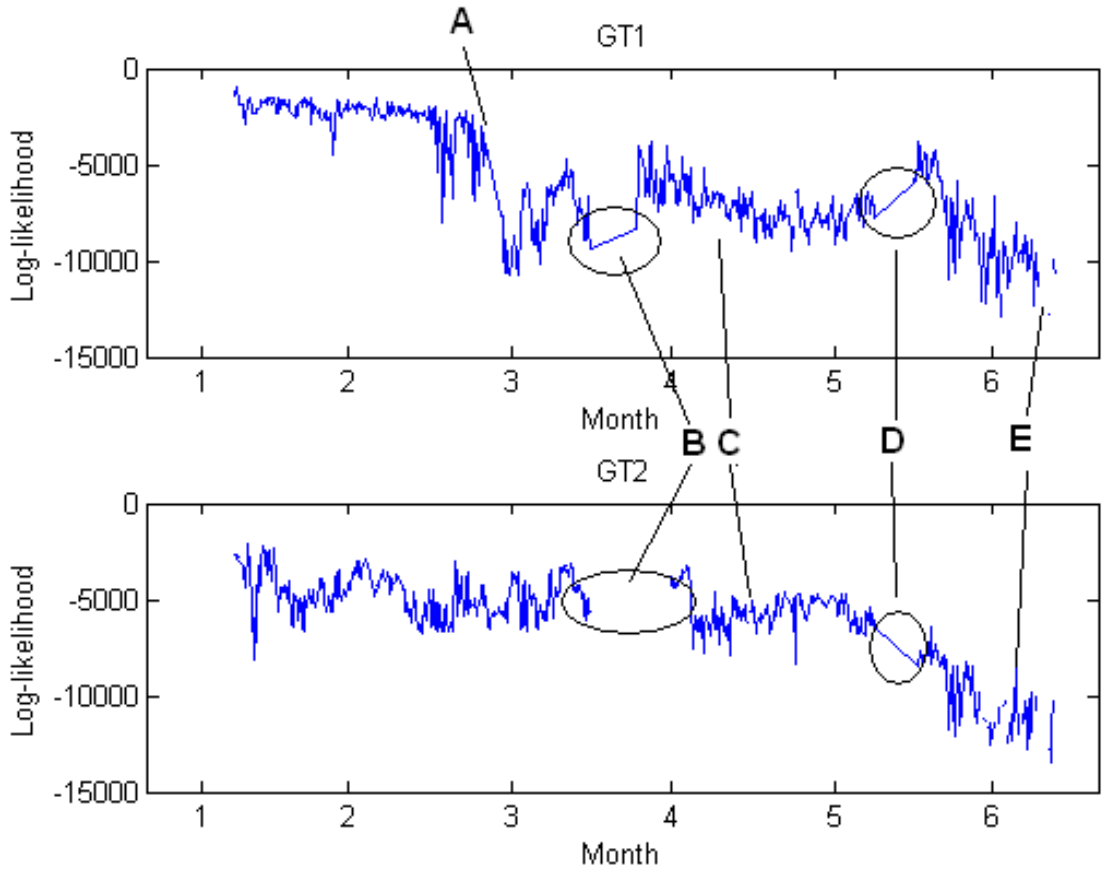

Figure 5.22: Initial test using MHMM on combustion data.

5.6.2 c Generalising the Model

As one of the advantages of HMMs is the ability to learn underlying behaviour, there is a possibility of deriving a general turbine model based on training data from multiple turbines. A model was trained using the combined training data of both GT1 and GT2. It should be stressed that this is limited to only two turbines of the same manufacturer and model, and as such further data and testing would be required to develop a true general GT model. In this case, it is useful because it provides a means to deploy a model for a known GT type even when no data is available to train a specific GT, such as when a new turbine is installed.

The results in Figure 5.23 show that the model with a combined training set produces results close to the individual models, particularly noticeable in the GT1 results. Interestingly, the general model actually shows a higher LL for GT2 than the GT2-specific

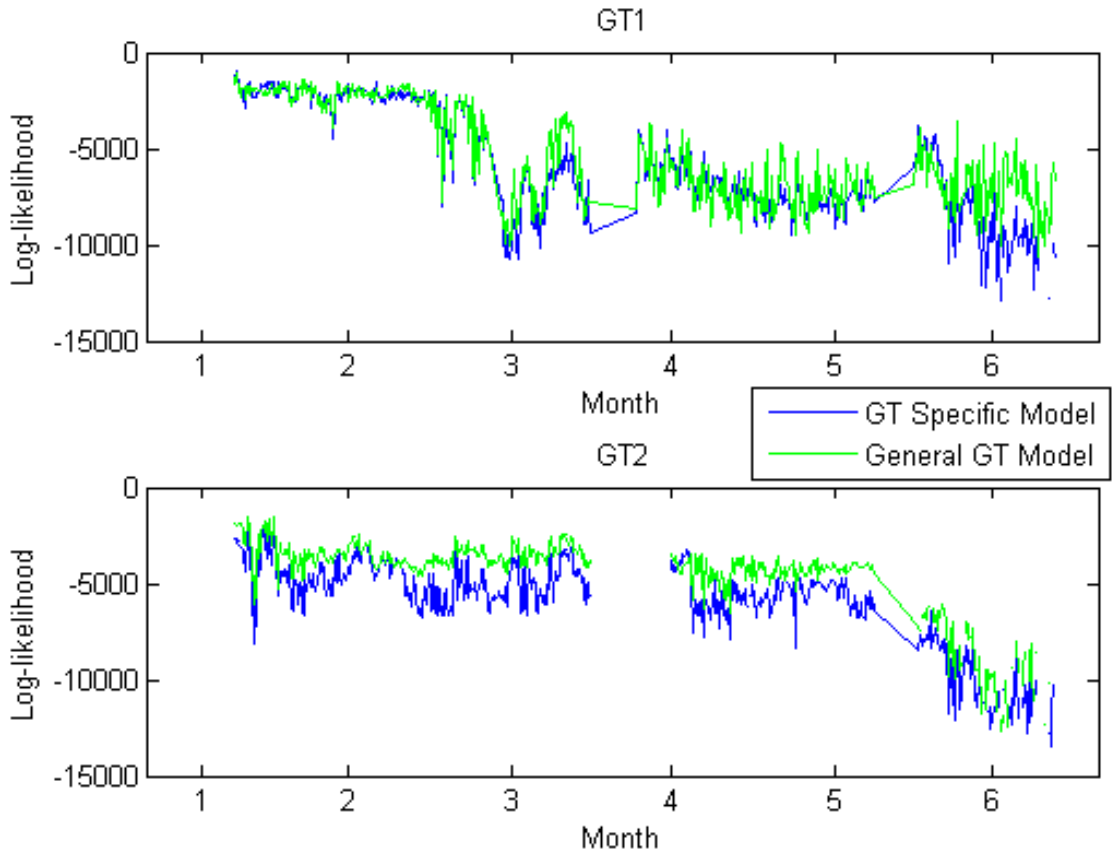

Figure 5.23: Comparison of GT-specific models to a model trained on data across both turbines.

model, but does still show degradation until the lower values towards the end of the time period. This would suggest that the dedicated model is more sensitive to the change in operational values due to water washes and other maintenance.

The results suggest that the general model is still able to continue effective anomaly detection. Despite the wider ranges of values, the general model has successfully learned the behavior of a turbine, even if the values vary slightly depending on the specific turbine. The use of a general model may make it possible to monitor a new turbine until sufficient data is available to allow a more sensitive turbine-specific model to be deployed. This approach shows promise and should be investigated further with additional data from other turbine makes and models.

5.6.3 Results

With the MHMM trained, its performance alongside the existing condition monitoring system was considered. The time and date of successfully detected anomalies were compared. The comparison focuses on major confirmed anomalies.

Table 5.4: Performance comparison of MHMM vs. SmartSignal.

GT	Month	Events	MHMM	SmartSignal	Similar
GT1	1	2	2	0	0
	2	2	1	1	0
	3	3	1	1	1
	4	1	1	0	0
	Total	**8**	**5**	**2**	**1**
GT2	1	2	1	1	0
	2	5	1	1	3
	3	2	2	0	0
	4	N/A	N/A	N/A	N/A
	Total	**9**	**4**	**2**	**3**

Table 5.4 shows a breakdown of the number of anomalies recorded during a four month period. The "MHMM" column shows the number of times the MHMM detected an anomaly before SmartSignal, or detected an anomaly which SmartSignal did not. The "SmartSignal" column counts the cases where the reverse is true, and the "similar" column counts the number of times where anomalies were detected within a similar time-scale. Data was unavailable for GT2 during month 4.

5.6.4 Evaluation and Comparison to Existing System

The HMM appears to reflect changes in the behaviour of the gas turbines, even with the relatively simple approach of taking the last LL value for each sequence. A side by side comparison shows the HMM outperforms the SBM-based system. This is particularly impressive as the training data contains only the original data - without the additional data added to the deployed model throughout the comparison period and beyond.

The results show that each system picks up different anomalies, with the MHMM identifying more than the existing system. While each individual system missed anomalies, no anomaly failed to be detected by one of the systems, suggesting that an optimal system would use both along with some means of corroboration.

This work focused on known significant incidents identified by station engineers during the period under study. It can be seen that all significant incidents were detected by at least one of the systems, confirming that the MHMM approach can present a useful complement to the existing system. These preliminary results show that the MHMM is capable of distinguishing normal and abnormal behaviour.

5.7 Applying HMMs to other subsystems

MHMMs were trained to the other variable groups (see Section 5.3) for GT1. Figure 5.24 shows the results.

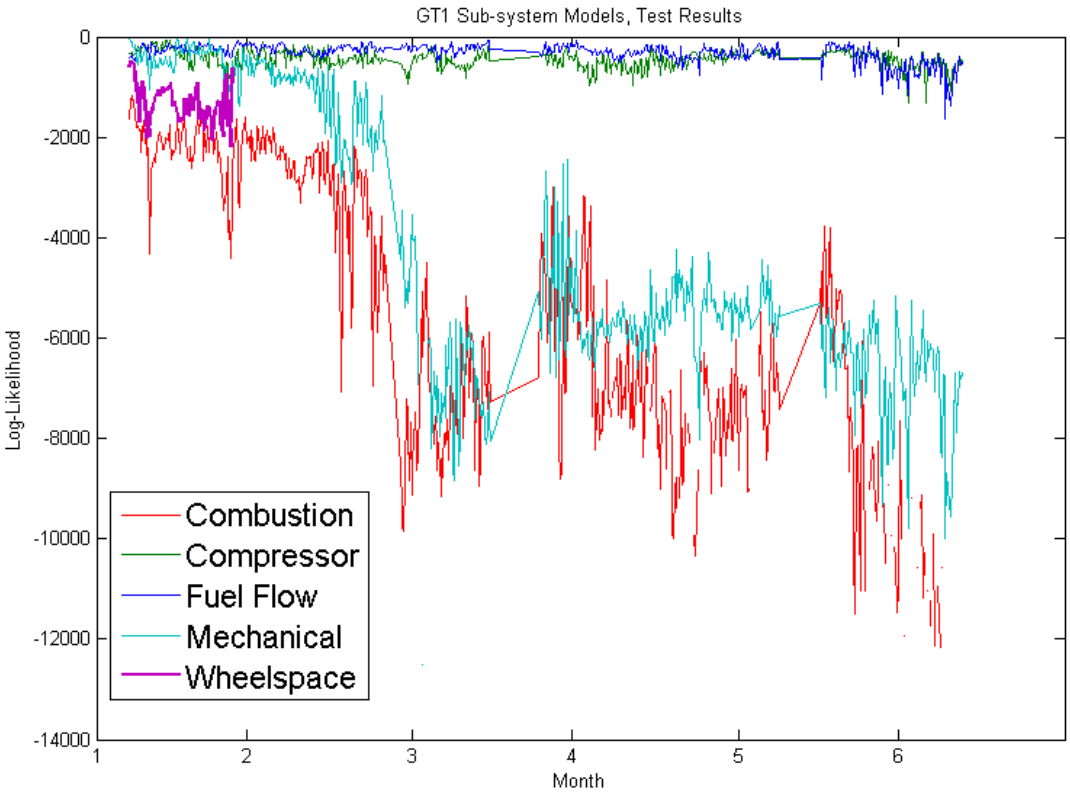

Figure 5.24: The log-likelihoods of all sub-system across the test period for GT1

The results show a number of interesting similarities between the sub-systems. The compressor and fuel flow appear display similar values. While the mechanical results starts with similar values to the compressor and fuel flow models, it drops following

the maintenance performed in month 2, at the same time as the dip in the combustion results (Figure 5.22, point A). From then on it displays results across the test period similar to the combustion model. The compressor and fuel flow models remain relatively unaffected by this maintenance.

The Wheelspace model returns values tending to negative infinity following the first month. This suggests an event that affected the wheelspace variables and changed the underlying behaviour and relationships. The change occurs at the same time as the second combustion event in month 1 (see Table 5.4). This suggests that while the combustion model picked up this anomaly, the effect on wheelspace was far greater and persistent.

The other subsystems all show a drop off starting in month 5, tending to negative infinity values as the turbine behaviour deviates significantly from the model.

These results suggest that training HMMs on other algorithms is practical, and can provide additional information about detected anomalies. This necessitates a mechanism to allow several HMMs to be deployed and interpreted concurrently. Chapter 6 proposes a multi-agent system to meet this requirement.

5.8 Automated Interpretation of HMM output

While section 5.6.3 shows the use of HMMs can be used effectively for condition monitoring, the results were only obtained through manual analysis of the log-likelihood values over the test period. This section provides a number of potential techniques to raise alarms automatically based on different characteristics of the LL outputs. A method utilising linear regression is also presented in order to track degradation of machine condition using the HMM output.

5.8.1 Invalid and infinite model output

The first potential indicator of an anomaly is when the output log-likelihood is so small it is unrepresentable. This is a result of an observation, state transition or the product of both resulting in a excessively small probability. The smallest positive number that may be represented, using standard 64-bit double precision floating-point representation, is approximately $2.22 \text{ x } 10^{-308}$ [149]. A positive probability value less than this is rounded down to zero and then produces a result of negative infinity when converted to a log

scale.

The probabilities are converted to log-likelihoods at each step of the sequence before being added to the previous likelihood. This means that the only way for a negative infinity value to occur is for a single observation to have an excessively low probability below the threshold for representation. If the probabilities over each sequence were calculated before conversion to a log scale, this would require several multiplications. This introduces the possibility of reaching the minimum representable value over the course of a long sequence despite the fact that individual observations may have far higher probabilities.

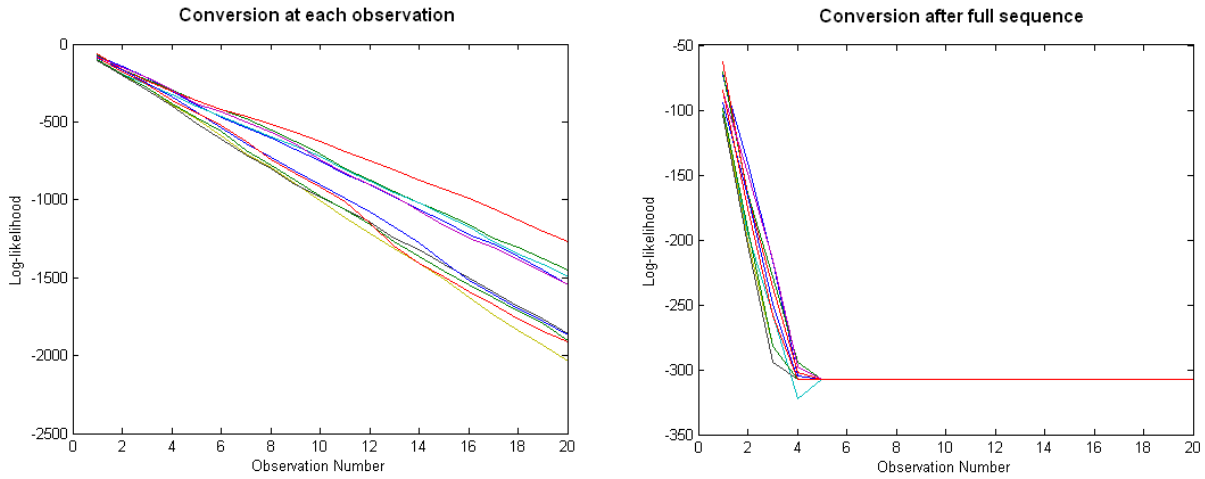

Figure 5.25: Comparison of approaches to conversion to log-likelihood values

Figure 5.25 shows a comparison of the same set of observation sequences. The left figure converts the probabilities to log-likelihoods for each observation before subtracting while the right figure shows what would happen if the entire sequence was converted after completion (with negative infinity replaced by the minimum possible value). Converting after completion is shown to reach the minimum double precision value after only a few observations.

The use of negative infinity values to indicate anomalies therefore sets a threshold of approximately -308 on any individual observation likelihood. This provides a clear and simple indicator of anomalies. As it only affects individual observations with extremely low LLs it should be useful in detecting the most severe faults and those which cause a sudden change in behaviour. However, as it is purely a yes/no form of fault detection, it can not be interpreted to provide additional information or an indicator of the extent

of the fault.

Figure 5.26 shows an example of several negative infinity LL sequences during a chosen test period. The example is from a mechanical model of a GT, over a 10 day period. The negative infinity sequences during the period correspond to known Mechanical anomalies on the 2nd, 3rd and 8th day.

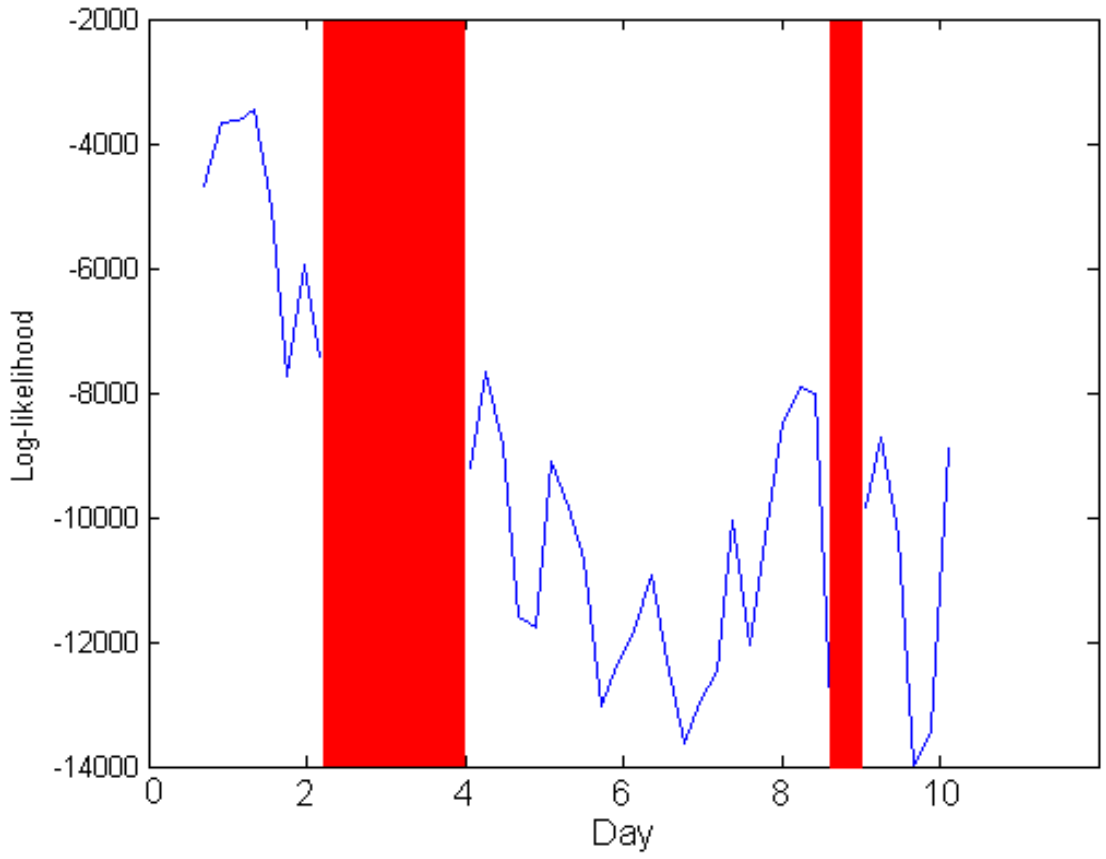

Figure 5.26: Example of negative infinity values as fault indicators. Data from a GT mechanical model.

While this is useful, sudden changes in behaviour can also be caused by transients. It is therefore a simple indicator, but one that may be prone to a large number of false positives.

5.8.2 Absolute log-likelihood value as measure of normality

The final value of a log-likelihood sequence has been used in previous chapters as the primary means to represent HMM output. This is because it provides a useful way to summarise the overall behaviour over a sequence, as the final value is the sum of all the individual likelihoods, and allows trending over time.

This approach proposes that the final value of log-likelihood be used directly as a measure for machine condition. This assumes that as machine condition degrades or faults present themselves this will be reflected in a lower log-likelihood value.

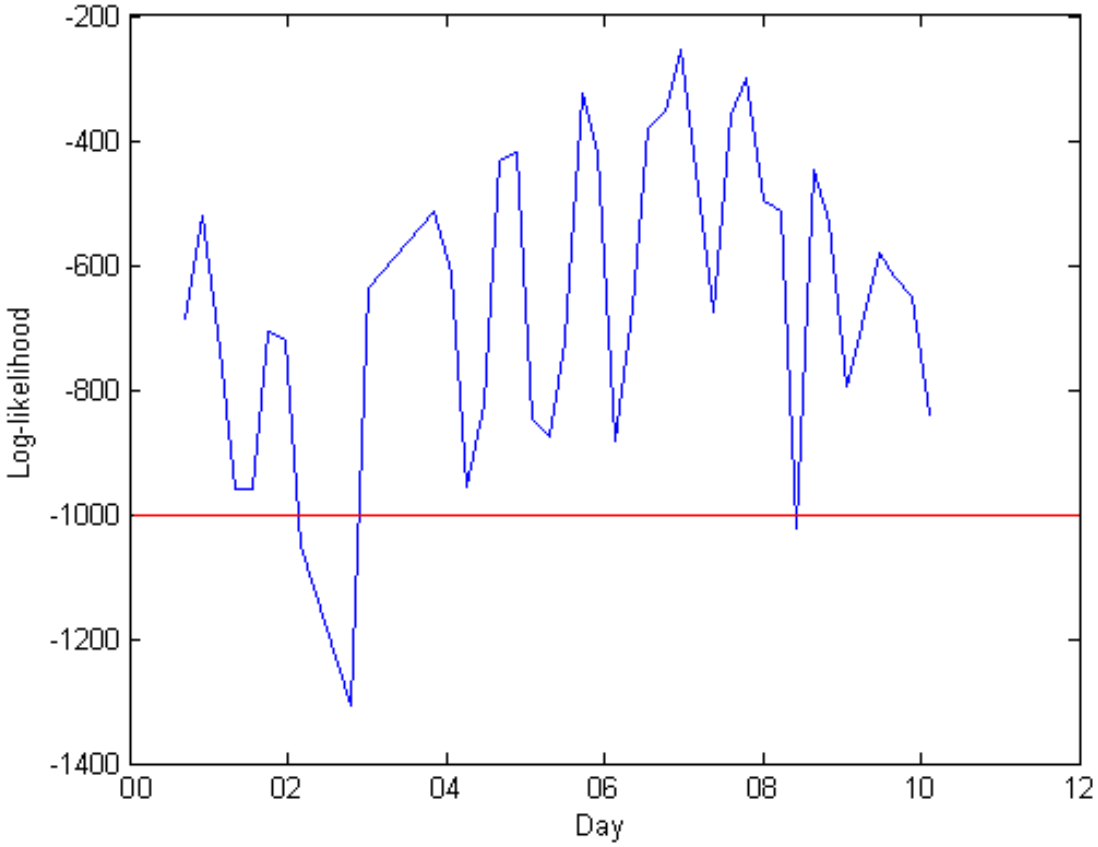

Figure 5.27: Using a log-likelihood threshold to faulty behaviour. Data from a GT mechanical model.

Figure 5.27 shows the absolute value assigned a threshold of -1000, with values lower than this threshold considered to be anomalous. The mechanical model is presented here, using the absolute log-likelihood with a threshold, correctly detects me-

chanical faults on the 5th day of the test period.

This approach has two disadvantages. The first is the need for a threshold to be set before testing. The example had a threshold set in order to show the potential to detect anomalies, but this was after already viewing the data. With some experience it may be possible for an engineer to set an appropriate threshold for future test data. A more flexible approach may be to base the threshold on training data, setting it slightly lower than the lowest LL output during the training period, as this represents "good" behaviour. Alternatively, known "bad" data could be input in order to determine the maximum threshold for detection of similar faults.

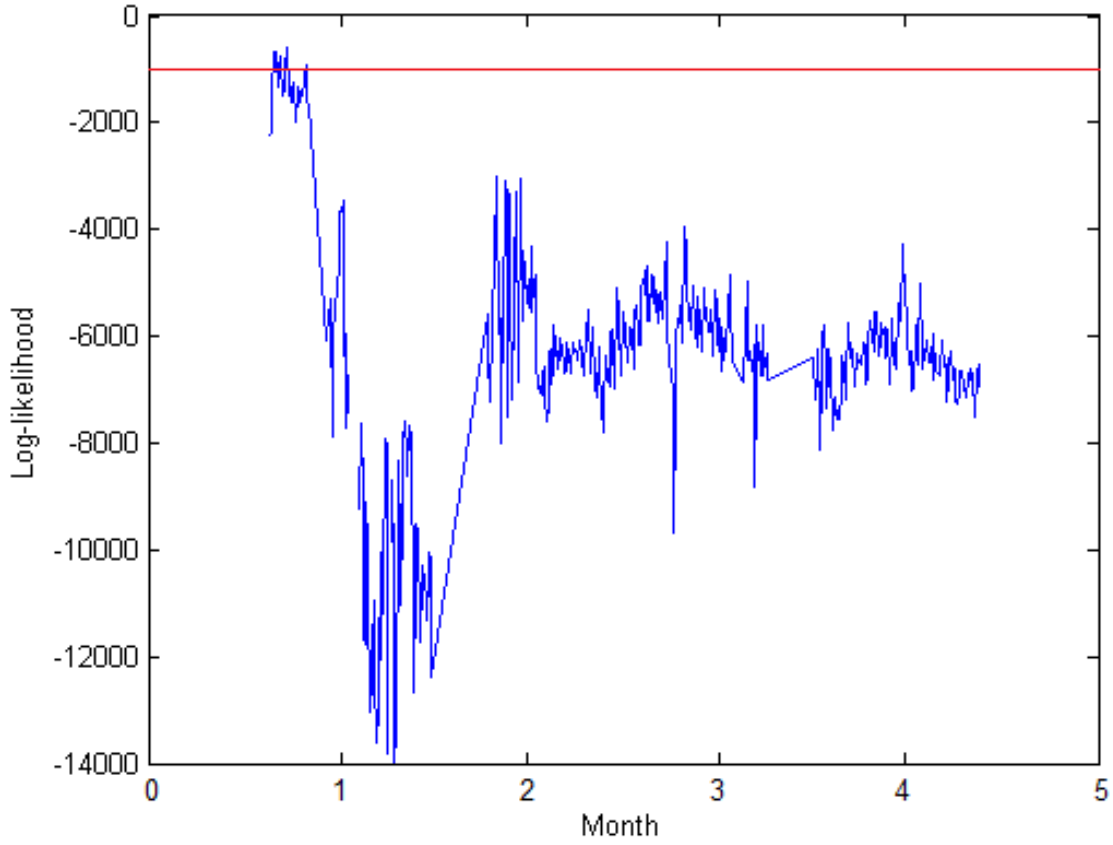

Figure 5.28: Example of high threshold value resulting in an excessively high number of false positives.

The second disadvantage to this approach is also linked to its most useful attribute: the LL value decreases with deviations in turbine behaviour from the training examples. While this is useful for tracking the wear and tear on a turbine and is useful as

an indicator of turbine condition, this makes setting a threshold for sudden and unexpected faults more difficult. Figure 5.28 shows that if the threshold used above to detect the mechanical faults were used for the rest of the testing period the model would flag anomalies for the remaining LL outputs.

While a lower threshold could be chosen to prevent the high number of alarms that would be raised from the results presented in Figure 5.28, this would prevent detection of the anomalies in Figure 5.27. The reason for the lower log-likelihoods after the faults is due to the change in behaviour resulting from either damage, or maintenance performed to address the faults. Even if the turbine is brought back up to as-new condition, its behaviour may not be exactly the same as before, resulting in a perceived difference from the model.

Changes in behaviour as a result of standard maintenance, such as water-washing, can be addressed by retraining. Genuine degradation would result in a lower LL value, as intended. However, this does mean than using the absolute LL value with a fixed threshold as the only detector of faults is not appropriate.

5.8.3 Rate of change of log-likelihood for anomaly detection

As the LL value of the sequences decreases due to machine degradation, sensitivity to sudden changes in behaviour can change as the machine degrades, especially when using a fixed threshold. In order to address this, a further measure that is sensitive to changes in log-likelihood, rather than the absolute value, is prudent. A comparison of the data from the previous example shown as both absolute value and rate of change is shown in Figure 5.29.

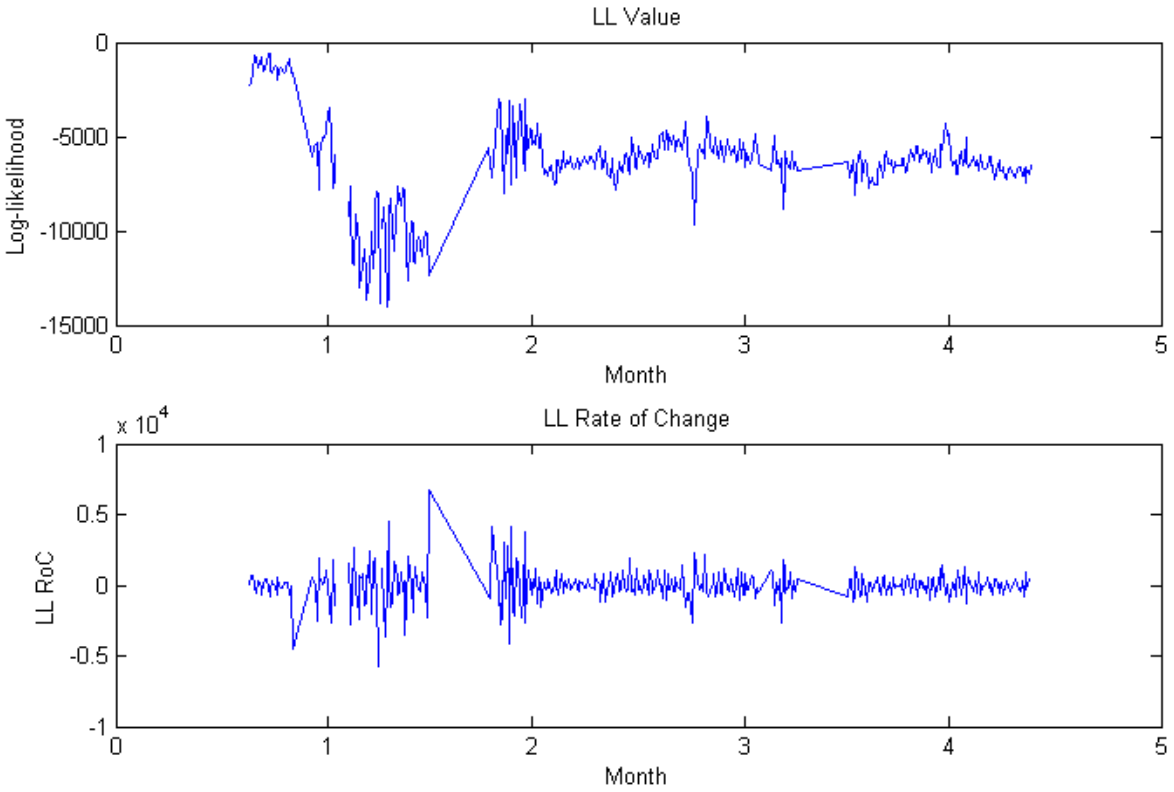

Figure 5.29: Comparison of Absolute Value and Rate of Change LL plots.

Using the rate of change, and setting the threshold to -2000, we are able to consistently detect faults throughout the time period. Figure 5.30 shows two large dips in the rate of change in months five and six. These both correspond to two periods of high vibration.

While the rate of change shows capability to detect faults, even when the machine already displays diminished performance, this also makes it less capable of tracking degradation of machine performance over time. However, this capability is provided by other methods. For this reason, this interpretation approach complements the other proposed techniques, and highlights the importance and robustness of using several methods.

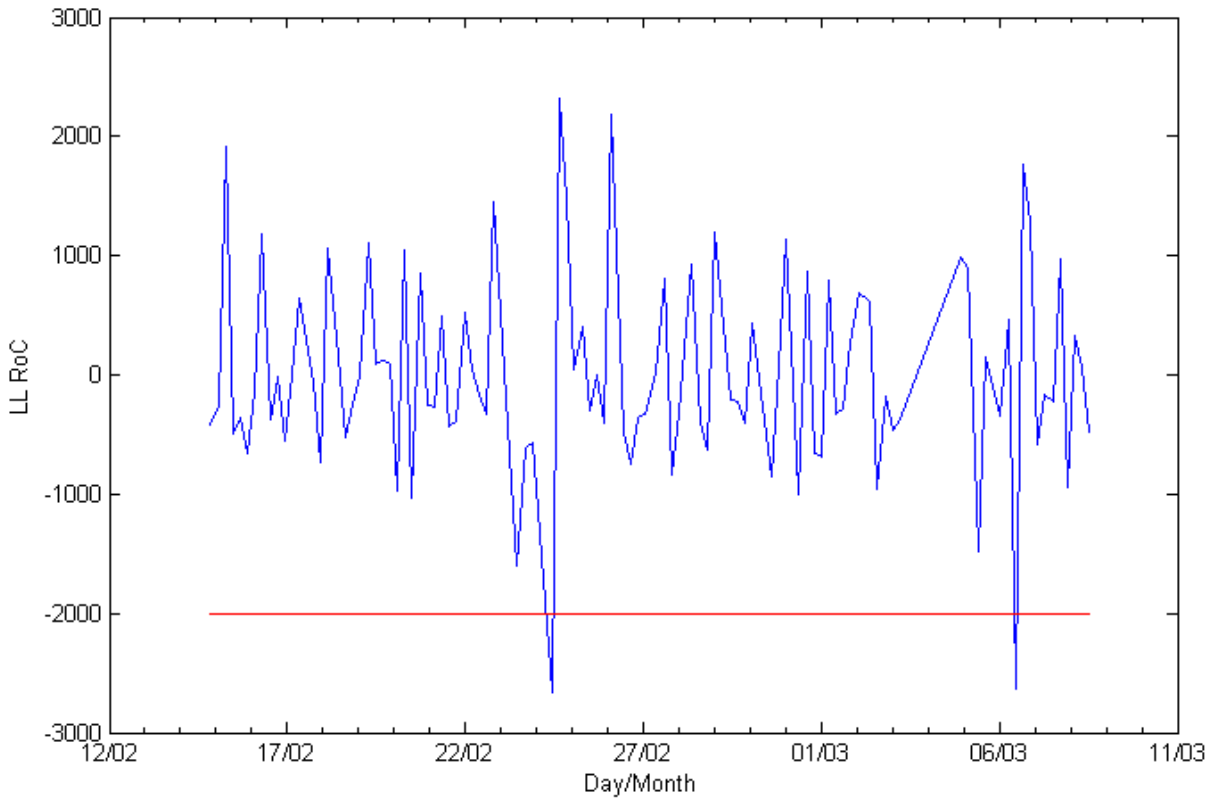

Figure 5.30: Rate of Change below threshold indicating two mechanical anomalies.

5.8.4 Variance of log-likelihood sequence as an indicator of fault

While the other techniques have considered only the final LL of a sequence, either checking it is finite (Section5.8.1), it remains above some pre-defined level (Section 5.8.2), or how it changes with time (Section 5.8.3), none consider the progress of LL over the entire sequence. While this simplifies the analysis, it leaves out potentially useful information provided by the model during duration of the sequence.

In order to provide a measure of the behaviour over a full sequence, without considering every observation, a statistical measure for the sequence was considered. Both mean and variance are useful measures and were considered. The mean was discounted as the mean is identical to the final log-likelihood divided by the length of the sequence, resulting in a scaled version of the information already provided by the approach considered in section 5.8.2. For this reason, the variance of the likelihood over a sequence was studied instead.

Figure 5.31 shows an example of a likelihood sequence compared to the average for

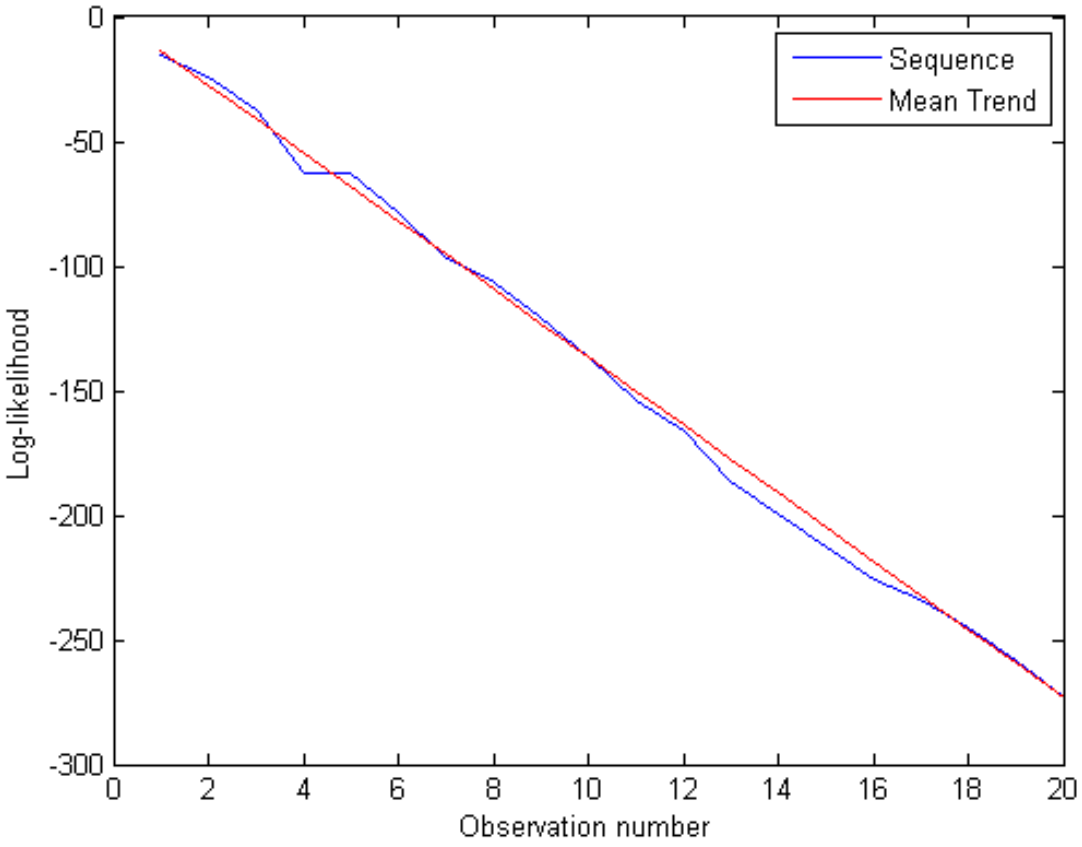

Figure 5.31: The actual likelihoods of a sequence overlaid with a trend of the mean gradient.

that sequence. The mean is represented as a diagonal line to reflect that LL is cumulative, each LL is the current likelihood in addition to the previous. The mean line is generated using equation 5.1. The variance is considered the square of the mean difference between this line and the actual value (equation 5.2).

$$\mu_i = i * y_I / I \tag{5.1}$$

$$\sigma = \sum_{i=1}^{I} (y_i - \mu_i)^2 \tag{5.2}$$

For equations 5.1 and 5.2 i is the observation number, I is the number of observations in a sequence, y_i is the likelihood of observation i, μ_i is the mean at point i, and σ is the

variance for the whole sequence.

When taking only the final log-likelihood of a sequence, it is possible that unusual behaviour over a few samples may be obscured by normal behaviour during the rest of the sequence. Figure 5.32 shows two sequences overlaid with the average trend.

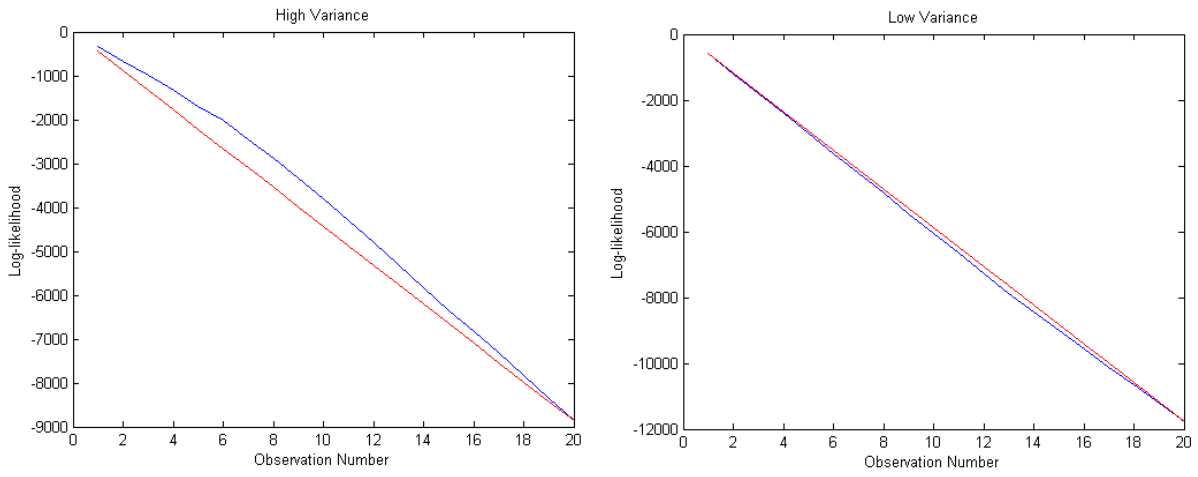

Figure 5.32: Comparison of two log-likelihood plots, with high (left) and low (right) variances.

The left plot in Figure 5.32 shows a high variance, with a clear deviation from the average. This appears to be caused by a relatively shallow trend at first, followed by a increased downward trend after about the 6th observation. This would suggest that the behaviour changed at this point, moving further away from the training examples that are considered normal behaviour. It is interesting to note that when compared to the right plot, the log-likelihood values are actually higher. This means that using only the final log-likelihood value (section 5.8.2), this change in behaviour mid-sequence would be missed, and examining the rate of change of sequence LL 5.8.3) would only detect such a change if the subsequent sequences continued to display a steeper LL trend.

While the variance of the sequences increases over time due to degradation (proportional to the decrease in average LL), the distance between genuinely anomalous behaviour and general degradation is still significant. With its focus on deviations from the average of the whole sequence, this is potentially vulnerable to anomalies caused by transients and should be considered alongside other measures.

Figure 5.33 once again shows the mechanical test set, with a threshold applied to the variance. This threshold was applied to the whole of the test set. The spikes are all clearly defined, and all but one correspond to a confirmed mechanical event, with the

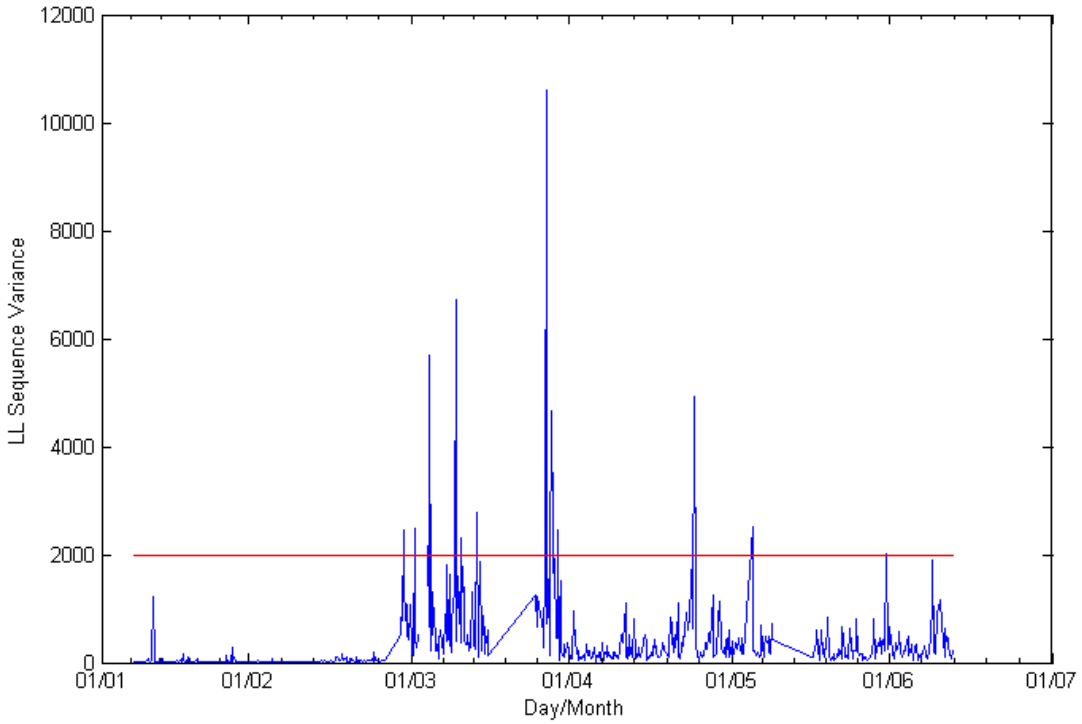

Figure 5.33: Threshold applied to the variance of mechanical model output.

remaining anomaly awaiting confirmation, showing a potentially high accuracy.

5.8.5 Trending Degradation of GT components

The LL value has been to shown to decrease as the observed behaviour deviates from the training data (section 5.8.2). However, it also showed brief "dips" when events occurred, whether genuine faults or transients, which we may refer to as "anomalies". This can make it difficult to perform either degradation tracking or anomaly detection due to the presence of both sets of information. While the other measures presented in this section show an ability to detect anomalies with a lower sensitivity to behavioural degradation reflected by lower LL values, a way to separate out the degradation information has yet to be implemented.

An additional issue can be the presence of noise in the output. The likelihood of an observation for each state is related to the probability of the observation across the probability distribution. If there is noise in the sensor reading used for the observation

vector, this can result in a change in the output from the GMMs and affect the output of the model. While using a log scale may reduce this variance in probability, it does not fully address the problem. Linear regression, in the form of the least-squares algorithm, is proposed to address both these issues and allow the trending of the degradation in turbine behaviour.

It is important to consider whether the data follows a linear trend, or whether a curve or other function may be more appropriate. The least-squares algorithm can be extended to other types of function. However, even in these cases it is often possible to convert to a form that allows the linear method to be used. In fact, this has already been carried out in the process of converting to a log scale, turning a probability curve into a likelihood line (see section 2.8.3, Figure 2.9).

While the least squares can approximate the trend in data where noise is present, it assumes that this trend is an uninterrupted straight line for the entire duration of sample data. However, due to faults and maintenance events resulting in sudden changes in behaviour, this is not the case. For this reason, an approach must be found to decide when such an event has occurred and restart the least squares process.

The approach used was to monitor the average absolute error between the fitted line and the actual values. If this increases consistently for a substantial period of time, this may suggest that the trend has changed and the process of line-fitting may need to restart. No analytical way of establishing the number of consecutive increases in average error was found. Instead, a process of trial and error was used and five (50 minutes) was found to provide acceptable results.

Figure 5.34 shows the linear regression process applied to combustion log-likelihoods. Two fitted lines are visible, both showing different rates of degradation. The second line shows a much steeper downward trend, suggesting an increasing deviation from normal behaviour. Around the time of the change, fuel flow issues are known to have been reported and were flagged as a potential issue. This would support the inference that the turbine's behaviour was changed.

It is important to note that although the LL increases during this period, the general trend is still downward at a similar rate. Despite this unexplained increase, the algorithm continues to use the same fitted line. This is considered a positive trait of the algorithm as it shows robustness to sudden changes that otherwise do not affect the fundamental trend.

The turbine continued to operate before dozens of events were reported followed

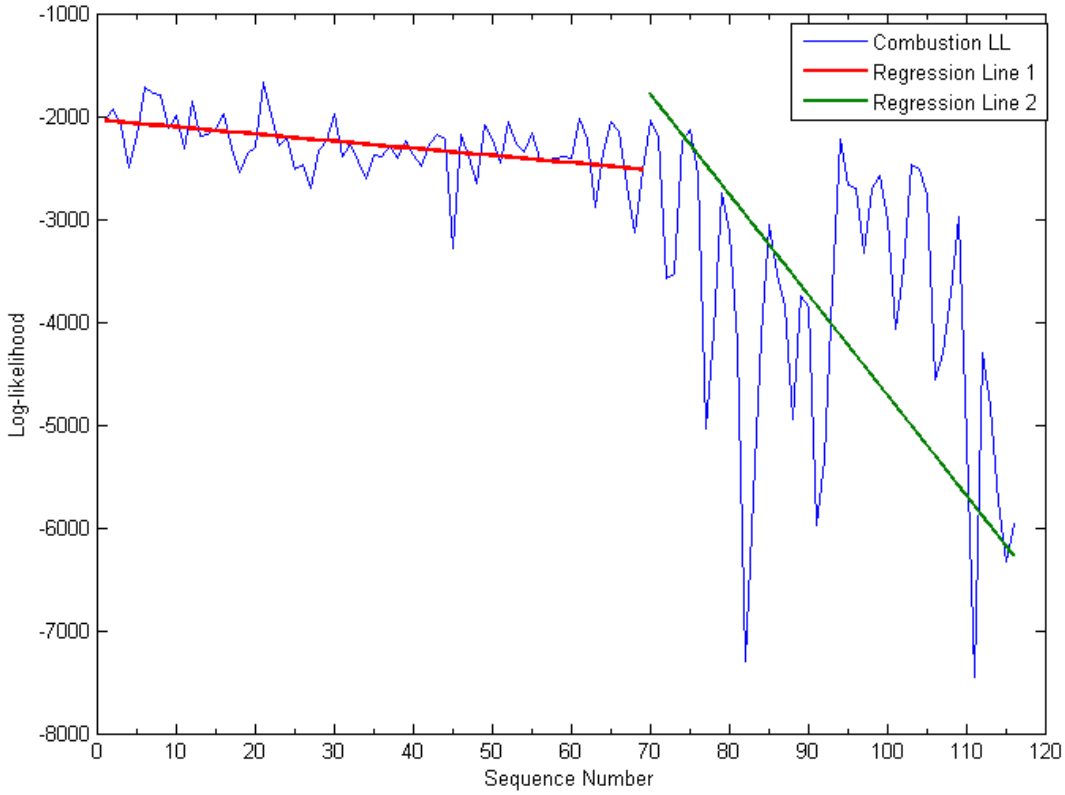

Figure 5.34: Example of least-squares line fitting applied to real LL output from combustion model.

by a shut-down (it is not clear whether this was a failure or a shut-down in order to investigate the anomalous behaviour). This suggests that the trending algorithm successfully detected increased degradation due to an anomalous event, that ultimately led to a serious failure.

5.9 Conclusion

This Chapter has shown that it is possible to capture the dynamics of a gas turbine despite the absence of high frequency data. It has presented two methods for the condition monitoring of gas turbines using low frequency data. An ANN-based approach has been shown to allow detection of exhaust gas anomalies in advance of failure, and potential capability for detection of more general faults. An MHMMs-based approach

shows particular potential, with the capability to successfully detect changes in the dynamic behaviour the sub-systems of the turbine. Specific examples relating to the compressor and combustion sub-systems show the technique to be capable of both anomaly detection and machine degradation tracking. Finally, this Chapter has presented a number of techniques for analysis of HMM results, which may be used in a automated condition monitoring system.

Chapter 6

Design of a gas turbine monitoring multi-agent system

The aim of the following chapter is identify the elements required for a flexible condition monitoring system, capable of modelling all sub-systems of a gas turbine. The system was required to allow parallel deployment of several models. The system also had to be extensible to allow new algorithms to be deployed as required. While any sufficiently modular software architecture can be extended to include new elements, a Multi-Agent System allows new modules to be added at run-time. This allows uninterrupted condition monitoring, with no re-compilation of code required. For these reasons, a multi-agent system was selected as the best means to deploy the algorithms.

The prototype system consists of a multi-agent architecture incorporating the HMM-based approach shown to be effective for several sub-systems in Chapter 5. The architecture is demonstrated to be extensible through the inclusion of an additional thresholding algorithm for each monitored variable. All of these algorithms are implemented as agents.

This chapter provides details of the agent architecture, including the types of agents and any communication between them. The ontology is an extension to an established power engineering ontology. A user interface for analysing the results is also present. Together, these elements fully realise a comprehensive GT condition monitoring multi-agent system.

6.1 Multi-Agent Architecture

The system is designed around a three level architecture, designed to allow extensibility through the addition of new functionality as agents.

Configuration These agents read the user-configurable set-up file and launch any appropriate algorithms or processing agents.

Algorithms The algorithms themselves. These analyse the data in some way before forwarding their findings to the processing agents.

Processing Agents designed to receive the results from the algorithm agents, process them, and store them in the database.

The structure of the system is shown in Figure 6.1.

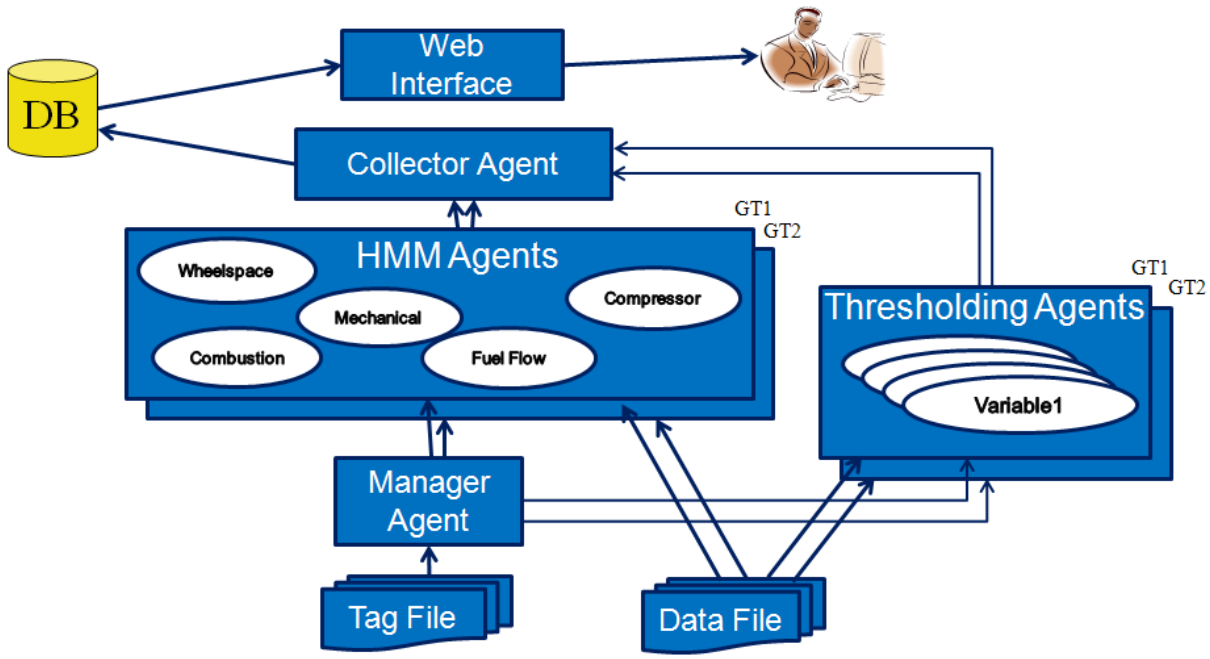

Figure 6.1: Architecture of prototype MAS

6.1.1 Configuration

A manager agent is used to dynamically launch the algorithm agents based on stored tag files. These contain the models and variables to be monitored, and are configurable by the user. This allows the user to launch appropriate agents for any given situation, and add agents while the system is on-line. In this way the system is highly configurable and adaptable.

The prototype is designed this way in order to allow the variables and models launched to be set by the user manually for the purposes of testing. However, a deployable system would be able to query the database in order to determine the appropriate models and variables to be monitored with no intervention from the user. This would allow the system to launch new agents as required. For example, if the database were expanded to include generator parameters, the system would automatically launch a generator HMM agent and any thresholding agents. In this way the system would provide a level of flexible autonomy.

6.1.2 Algorithms

An agent implementation of the HMM algorithm was created, allowing as many models to be trained and deployed as the system requires. In addition to this, a simple thresholding algorithm was implemented, permitting a threshold agent for each monitored variable. This was trained on the same data as the HMM and set limits on normal behaviour values. If a variable exceeds these limits an anomaly would be flagged. For the prototype, five HMM Agents (One for each of the variable groups described in Section 5.3) and 120 thresholding agents (one for every variable) were deployed for each turbine.

6.1.3 Data Management

In addition to these two algorithm-based agent types, a collector agent was used to store the result from the other agents to a database. Because the collector agent is the only agent in the system to communicate with the database, the system is highly flexible. Depending on the users' preference of database it is possible to create and deploy an appropriate collector agent, without having to rewrite or modify any existing algorithm agents, reducing the work to adapt to a new system or schema.

6.2 A Condition Monitoring Ontology

Rather than build a new ontology from scratch, an existing ontology was sought that might fit the requirements of the application. This was found in the power systems upper ontology [150] developed by the IEEE Power Systems Working Group.

6.2.1 Power Systems Upper Ontology

The upper ontology was developed following experience in integrating the COMMAS (Section 3.5.3) and PEDA (Section 3.5.4), where it was found that ontological issues were a major stumbling block [151]. The upper ontology is designed to incorporate terms common across power system applications [152], in order to speed development and ensure high-level compatibility and communication is possible between agents of the same domain. The upper ontology is then supplemented by a lower ontology that fills in the low level detail necessary for the specific task.

The upper ontology is based upon the Common Information Model (CIM) [153] for communication between energy management systems. This ensures the terms used are familiar and transparent to utility engineers. The implemented CIM terms are shown in Figure 6.2

The terms from the CIM shown in Figure 6.2 fall under the term “Concept”.

The terms “Concept”, “Predicate” and “AgentAction” are a requirement for JADE messaging. All terms in the ontology fall under “Predicate” - an expression that can be evaluated true or false, or “Concept” - any other physical or abstract thing or function. The included predicates are shown in Figure 6.3. These two terms make up the highest level of the ontology, while “AgentAction” is included as a sub-term of Concept.

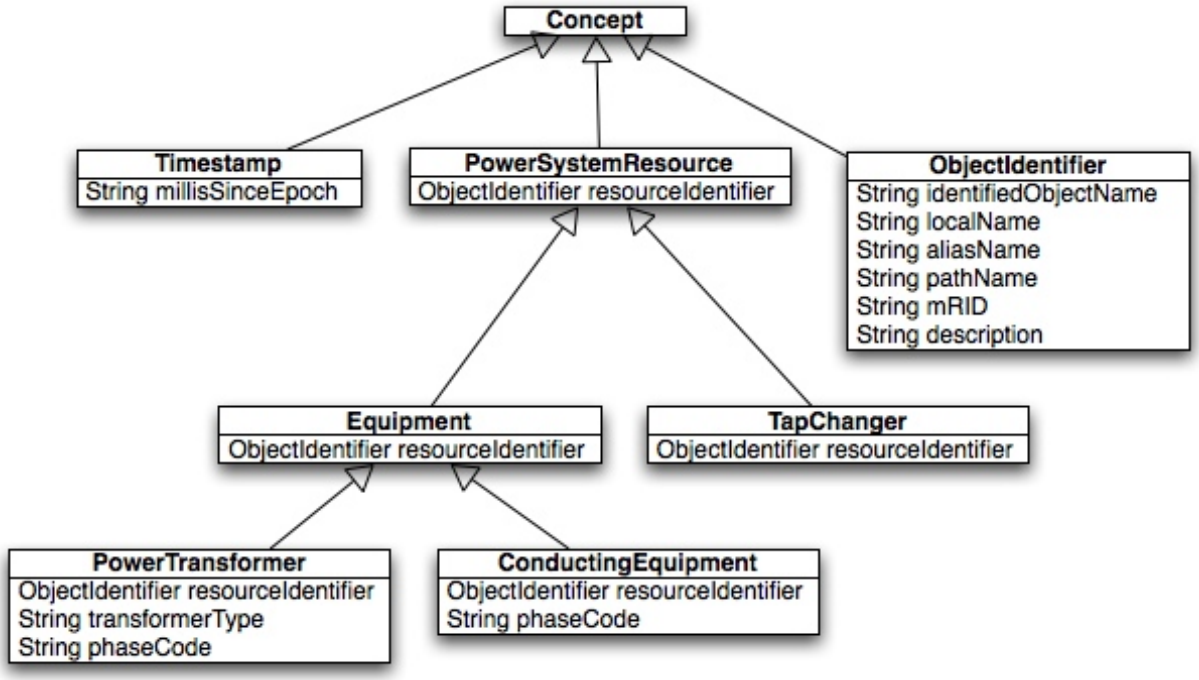

Figure 6.2: CIM concepts implemented in the upper ontology [150].

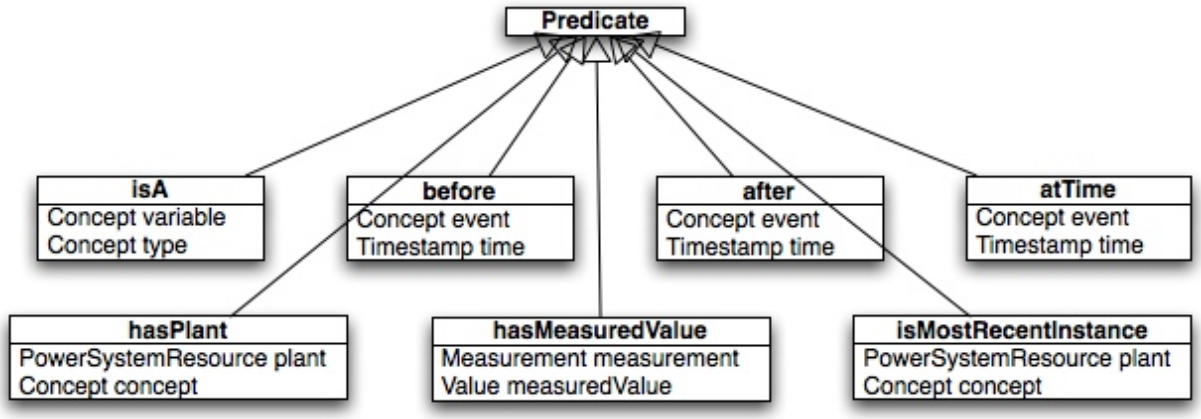

Figure 6.3: Predicates included in the upper ontology [150].

6.2.2 Extensions to the Upper Ontology

While the upper ontology forms a useful vocabulary of power system terms, it cannot fulfil the requirements of an ontology for the proposed MAS without extension. This is due to the nature of the ontology - it is only intended to be a high level vocabulary, and is not intended to completely define any particular application ontology.

The first addition to the ontology was the inclusion of an observation concept. This includes the one or many measurements that make up an observation. Its place in the ontology is shown in 6.4. Multiple instances are used to represent the observation sequences used to train and test a HMM.

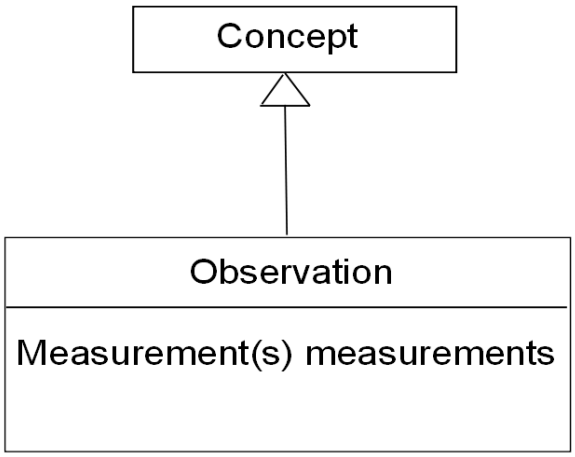

Figure 6.4: The "Observation" concept and its place in the expanded ontology.

It was also necessary to add concepts to represent the log-likelihood output of the HMM. This was implemented as a sub-concept of value, (figure 6.5). It takes all its attributes from its parent concept, with the constraint that the "value" attribute must be of type float. It should also be noted that the "unitSymbol" is currently not used as we are dealing with probabilities and therefore a unitless value.

A "LogLikelihood" often belongs to a LL sequence, and this had to be represented in the ontology as well, as agents communicating model behaviour over a historic period may require to send information about many LL sequences.

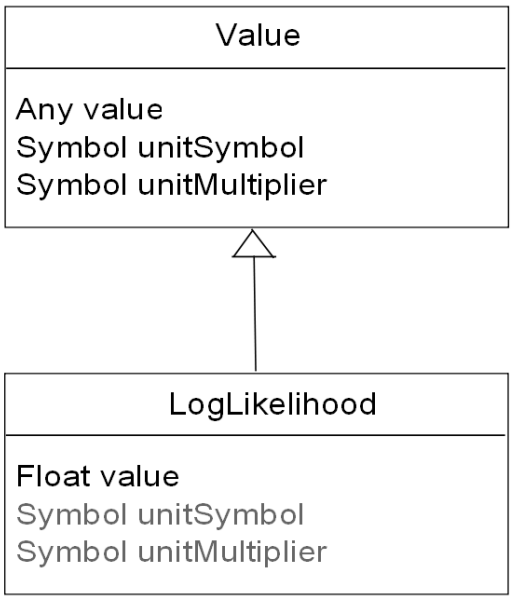

Figure 6.5: The "LogLikelihood" concept and its place in the expanded ontology.

In addition to the LL of a sequence of observations, an HMM may provide the capability to calculate the most probable state sequence. This can be useful when attempting to understand the underlying behaviour of the turbine. While this functionality was not utilised in the work presented in this thesis, it may be added in the future, and therefore it is prudent to include this concept in the ontology. Both sequence types are children of "DetailedInterpretation", which is in turn a child of the "InterpretedData" concept, as shown in Figure 6.6.

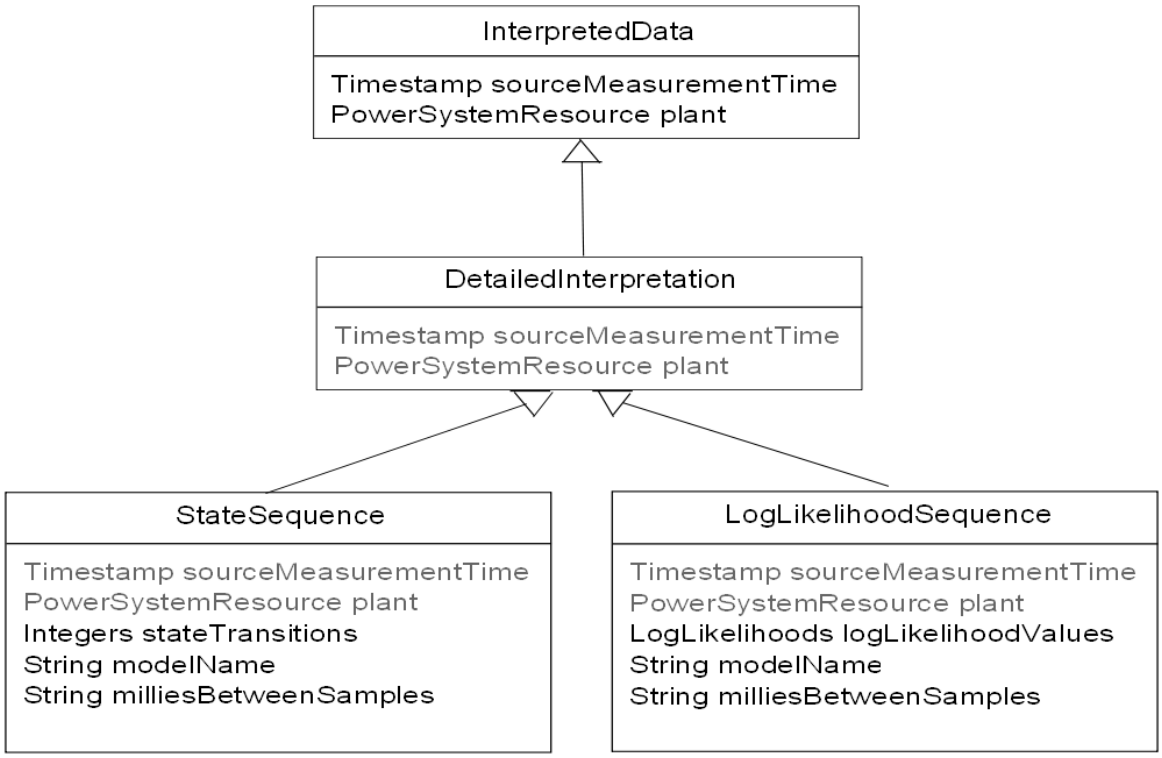

Figure 6.6: The LogLikelihoodSequence and StateSequence concepts and their place in the expanded ontology.

6.3 Agent Operation and Communication

With the agents and their roles now defined, along with the required ontology, it is possible to analyse the communication that occurs between agents that allows the multi-agent system to function as a single condition monitoring system.

6.3.1 Launching a set of agents

The process of initialising a set of agents to monitor a particular gas turbine begins with launching a ManagerAgent. This agent takes a single argument: the location of a folder containing several items related to the operation of the system:

Variable Tag File	A CSV file containing a list of the variables the system is to monitor, and the models to which they belong.
Data File	The Test data. By changing this file the user may run many different test sets through the same model.
Configuration File	Contains settings relating to data processing and setting up a database connection.

All the above files are human readable and editable, and are intended to be. This allows a user to easily deploy custom sets of agents as they require.

At the moment training is carried out off-line by a separate Java program, but if required this could also be performed by agents. The post-training information is stored in a sub-folder which contains the following:

Trained HMM files	Pre-trained models which are used during the testing. One for each MHMMAgent
Threshold file	The anomaly thresholds for each variable, set during training. One entry for each ThresholdAgent

While the prototype follows a set directory structure to speed development and testing, a full system could be extended to allow the user to select training and test files through a standard GUI interface. The manager agent launches an MHMMAgent for every trained model file found. A ThresholdAgent is also launched for every tag and threshold pair in the Threshold File. Once the ManagerAgent has confirmed with the DF that the appropriate agents have all successfully registered their services with the platform, its task is completed and the ManagerAgent disconnects from the platform and terminates.

6.3.2 Testing using a MHMMAgent

The location of the HMM training file is passed as an argument, from which the agent initialises a MHMM. The manager agent registers its service descriptor with the DF.

The file containing the test data is also passed in as an argument, allowing the agent to immediately begin testing. At this point the data is provided only via a dump file from a plant information (PI) database. However, in the future it is likely that on-line data may be used, or other data sources may be used. For this reason it is intended, in future, to extend the system to include a data extraction agent, to provide a level of abstraction and adaptability to for various data sources.

The MHMMAgent parses the data file, processing the data into observation sequences before generating results in the form of LogLikelihoodSequences. These are passed to any subscribed CollectorAgent.

6.3.3 Anomaly Detection with a ThresholdAgent

The threshold agent is initialised with the variable to be monitored, the threshold value, and the test data file set as arguments. Once registering itself and its services with the platform, it begins testing on the provided data. The thresholding agents are simple yes/no agents which mark a measurement as anomalous if it exceeds the indicated threshold. This is currently set at the maximum value encountered during training. A CollectorAgent may subscribe to the ThresholdAgent to be informed of any classified measurements as they are processed.

6.3.4 Result collection and storage

A single database connection is maintained by the CollectorAgent. This simplifies database access, avoiding issues associated with concurrent read/write operations. The collector agent subscribes to the DF to be aware of any agents, now or in the future, offering anomaly or log-likelihood sequences as services. It then subscribes to the appropriate agents in order to be informed of any newly processed measurements (whether they are classified as anomalous or not) or LL sequences. Upon receiving an "Inform" ACL message it saves the information to an SQL database for further analysis and visualisation.

A diagram of the interaction between agents is shown in Figure 6.7.

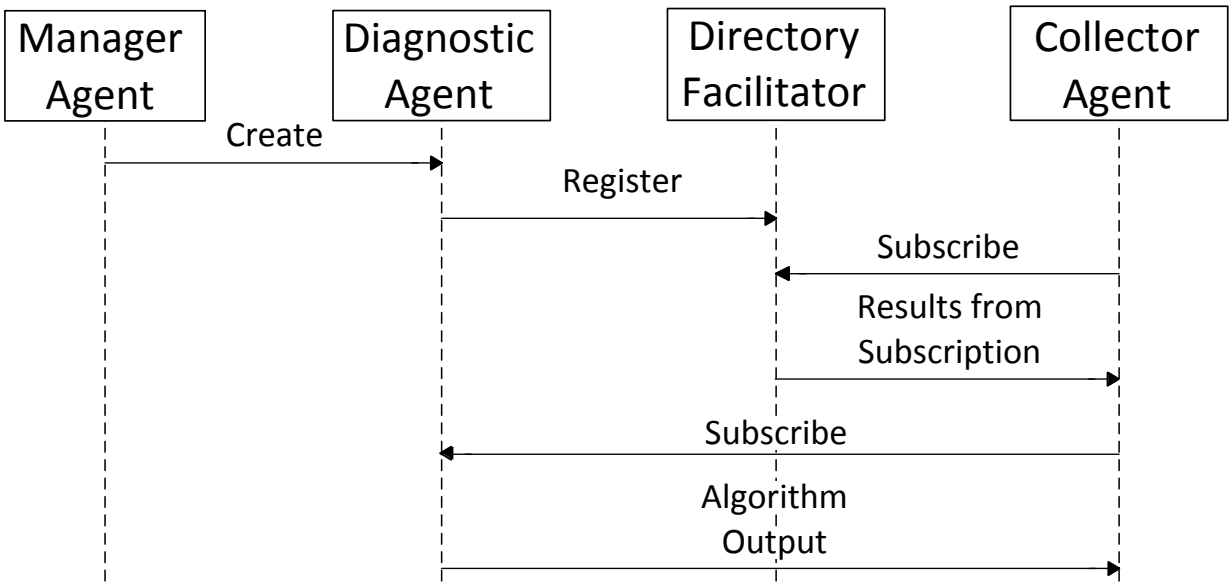

Figure 6.7: Interactions between the DF, ManagerAgent, CollectorAgent and a single algorithm agent.

6.4 Database Design

A MySQL database is used to store all relevant variable(tag), measurement and LL data. The database allows data from across many plants to be stored, potentially allowing cross-comparison of different plant items.

6.5 Viewing the Results

To display the results, a custom web interface was developed. The use of a web interface allows the user interface to also be platform independent and ensures that anyone within the utility can view the results of the system. Figure 6.8 shows the web interface presenting the results from the five HMM models for one turbine.

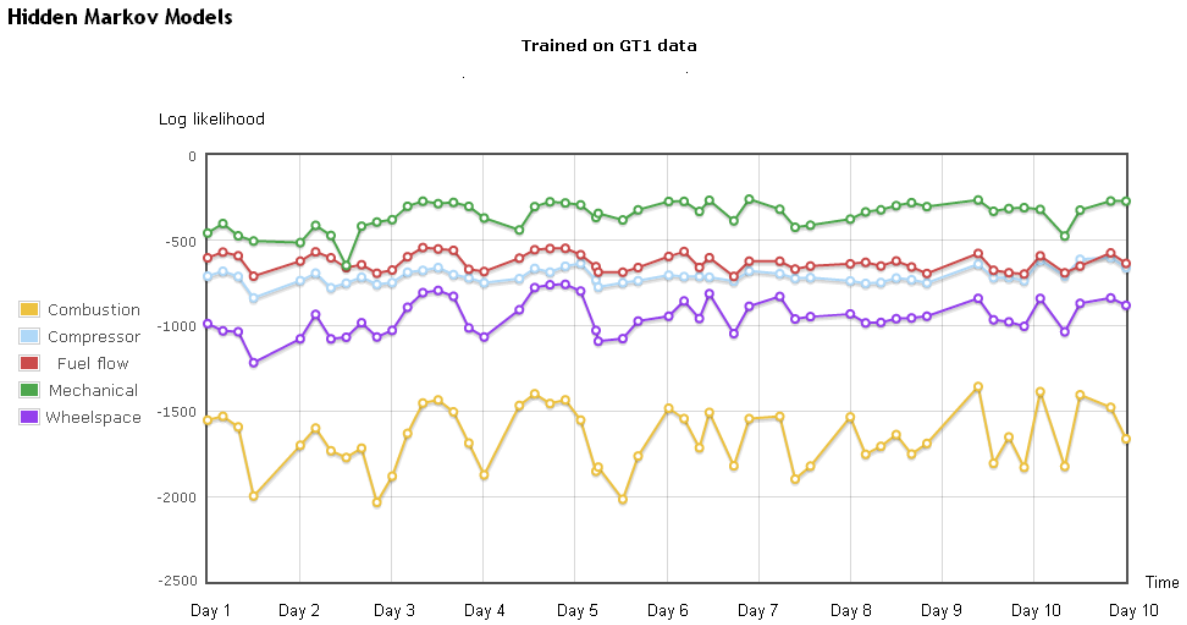

Figure 6.8: Results from all five GT HMM models

The interface also provides the ability to focus in on a number of LL sequences and view the log-likelihood of the observations as the sequence progresses in a manner similar to the approach used previously by Fox et al [56]. This is provided for any of the deployed HMM, as shown in Figure 6.9.

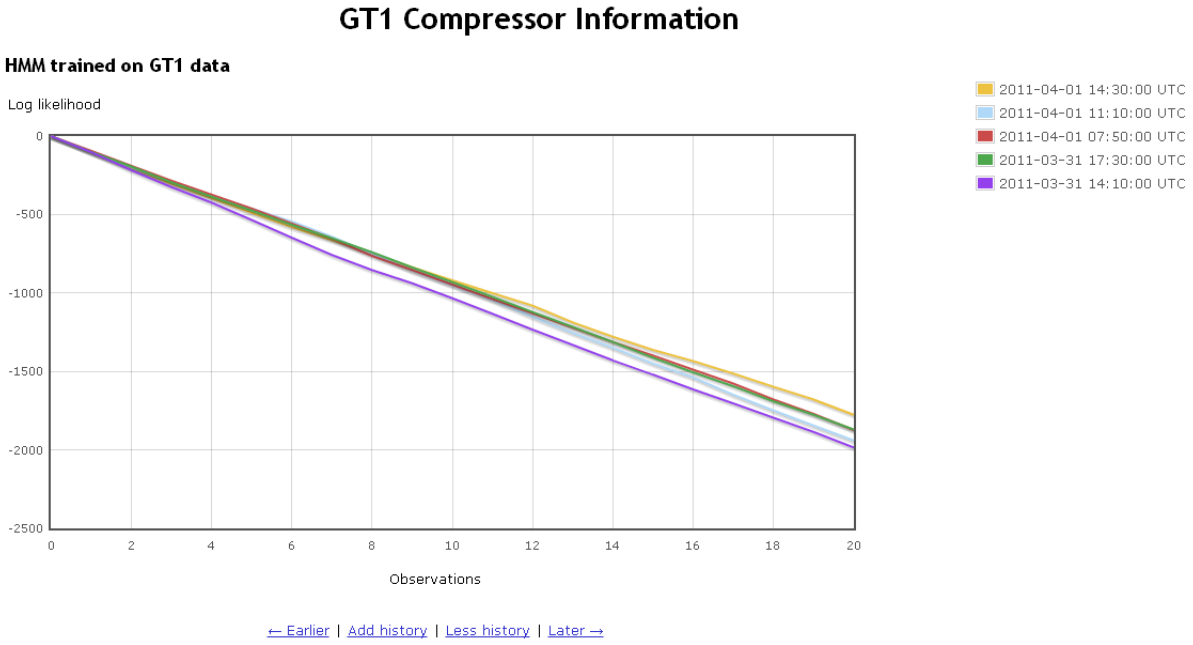

Figure 6.9: A subset of five log-likelihood sequences displayed in the web interface.

The web interface also provides a means to plot the anomalies found by the thresholding agents of several variables concurrently (shown in Figure 6.10). This is available on the same page as the sequence plot and is presented alongside a plot of the offending variable. This provides a useful visualisation tool and helps in matching up the behaviour of the HMMs and the variable values with the detected anomalies.

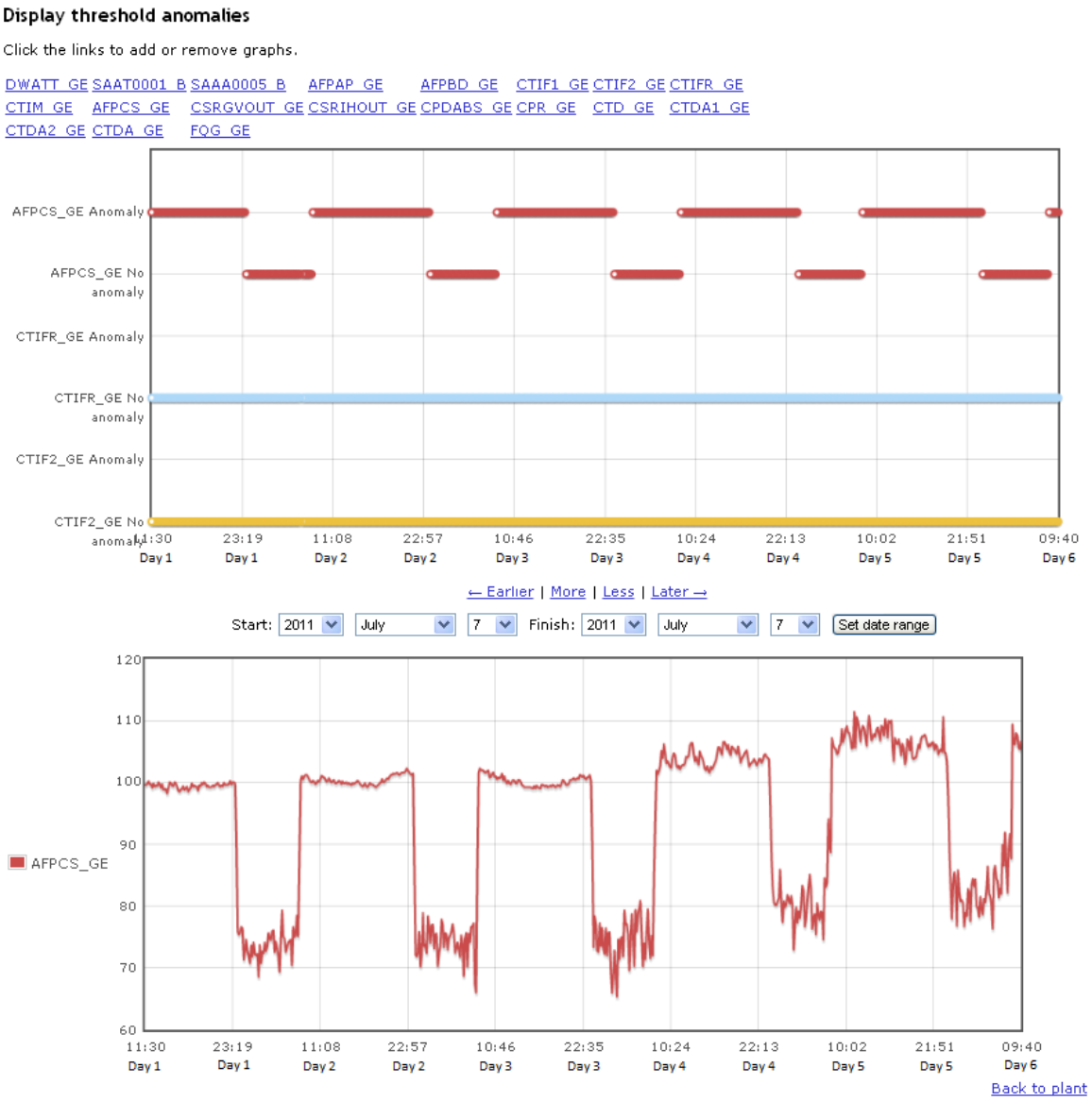

Figure 6.10: View of threshold anomalies alongside variable plot.

The combination of MAS, database and web interface provide a condition monitoring platform containing multiple models and algorithms for multiple GTs.

6.6 Demonstration

The following is a demonstration of the MAS prototype, designed to show its capability to monitor multiple models and variables, which also retaining the capability to react to additional information, such as new models or additional parameters and thresholds.

6.6.1 Launching the Agent Platform and Monitoring Combustion

The agent platform is first launched, containing only the Agent Management System (AMS), Directory Facilitator (DF) and Remote Monitoring Agent (RMA). A ManagerAgent is then launched, configured to launch only a combustion model and thresholding agents for combustion related parameters, before terminating. This is shown in figure 6.11.

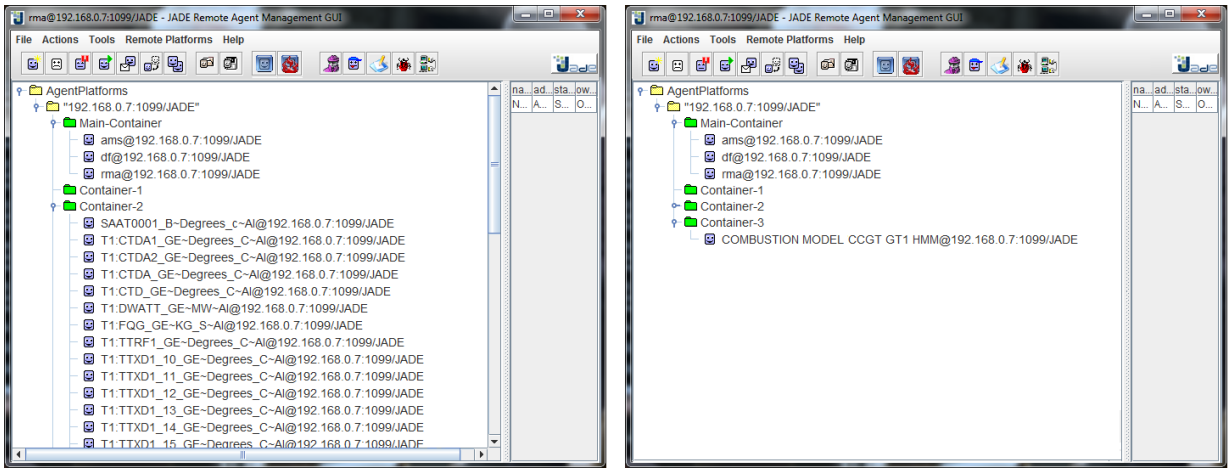

Figure 6.11: The JADE GUI showing the combustion agents following launch. Thresholding agents in Container-2, as shown on the left. Right shows Container-3 contains the HMM agent.

6.6.2 Collecting and Viewing the Data

With the agents now launched and monitoring combustion, it is necessary to store the results in the database. A collector agent is launched to carry out this task, as shown in Figure 6.12.

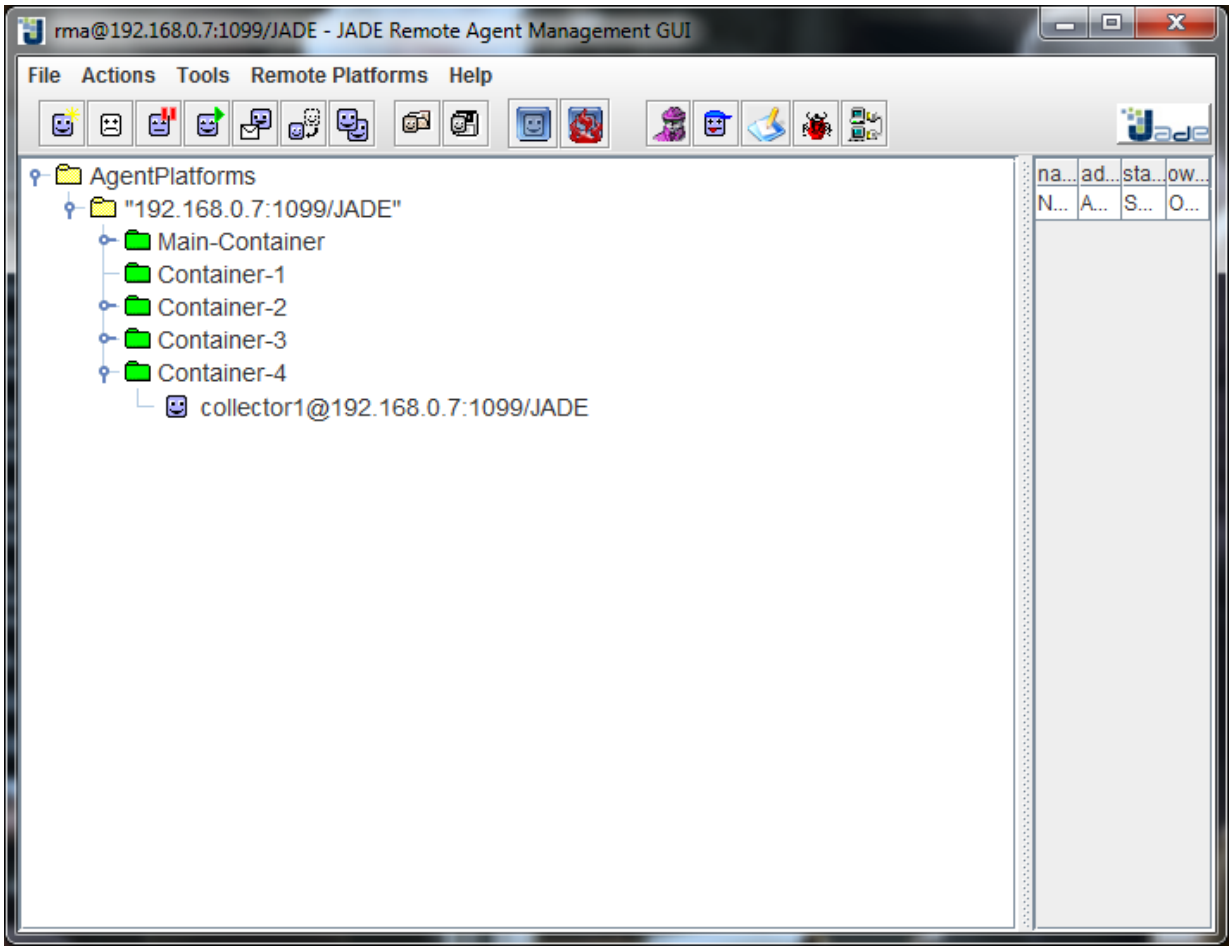

Figure 6.12: The JADE GUI showing the collector agent successfully launched in order to store combustion CM results to the database.

With the results now being stored in the database, it is possible to launch the web interface and view them, as shown in Figure 6.13

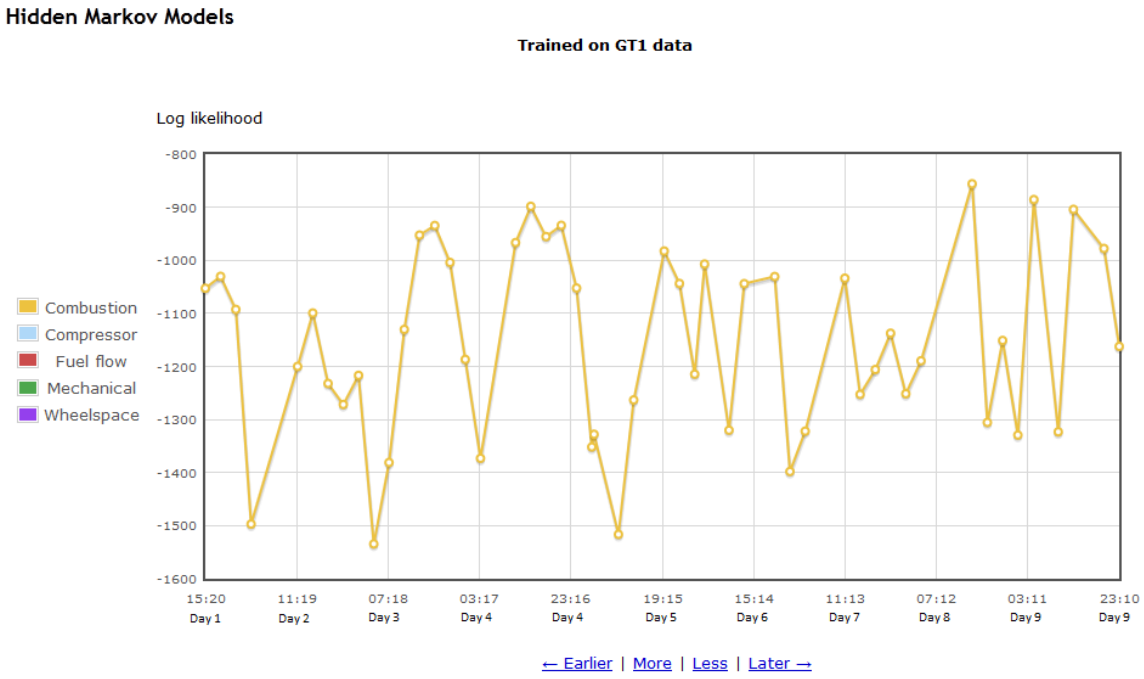

Figure 6.13: The web interface showing log-likelihood results from the combustion HMM only.

6.6.3 Monitoring Additional GT Sub-systems

The system will now expand to allow monitoring of other sub-systems of the gas turbine. The models and parameters for the compressor, fuel flow and mechanical sub-systems will now be deployed by launching another ManagerAgent configured to launch the appropriate agents. Figure 6.14 shows the JADE GUI with the new agents running.

If we refresh the web interface it is possible to view results for all 4 of the sub-systems, as shown in Figure 6.15.

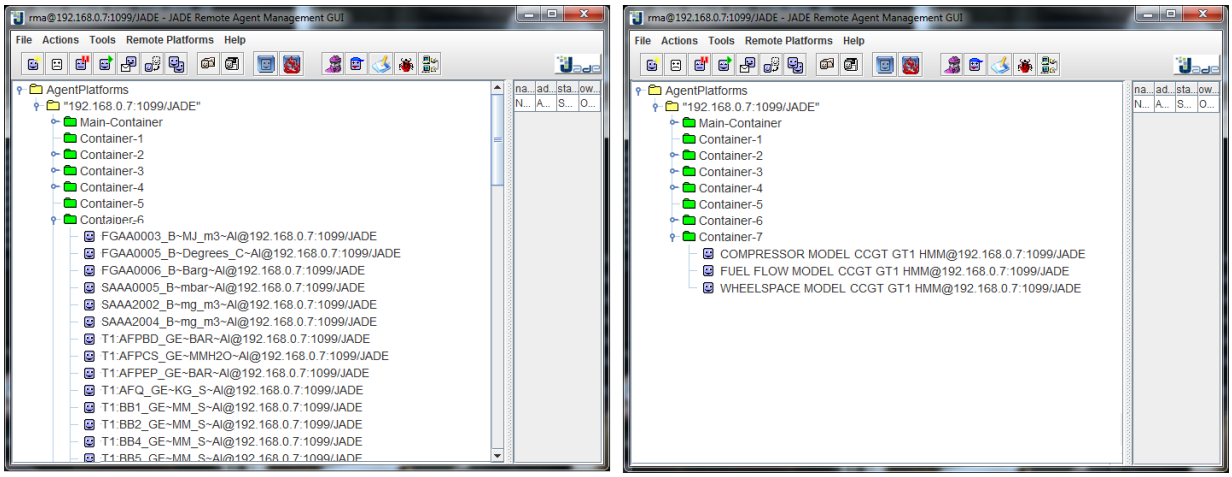

Figure 6.14: The JADE GUI showing the additional agents deployed. Thresholding agents in Container-6, as shown on the left. Right shows Container-7 contains the remaining HMM agent.

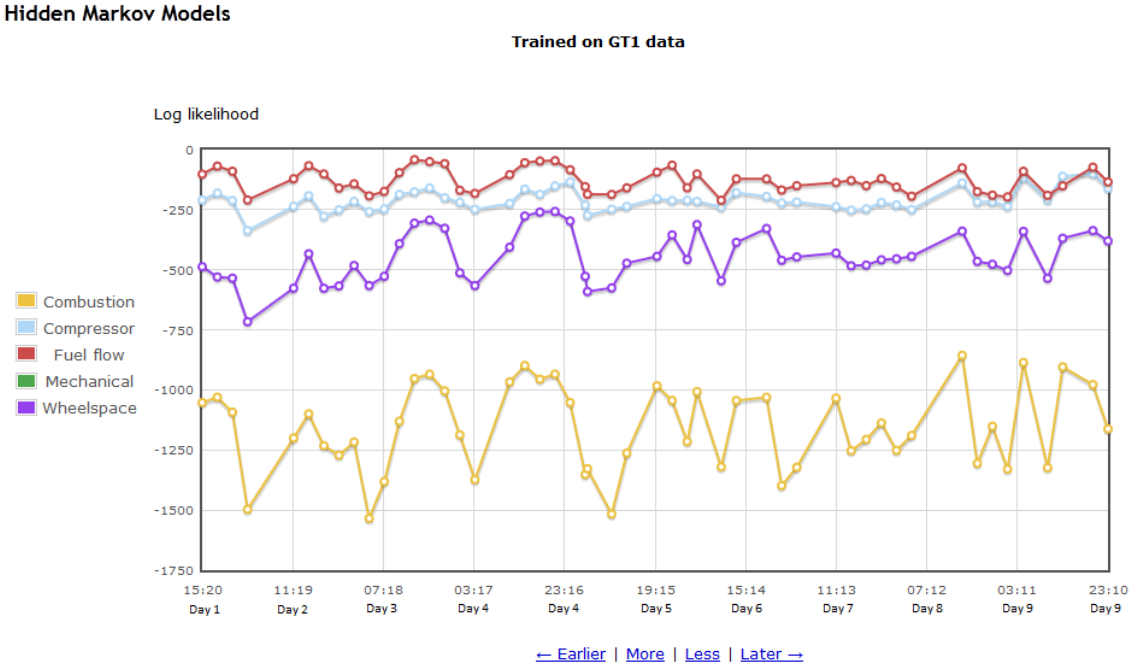

Figure 6.15: The web interface showing log-likelihood results from four of the subsystems.

This demonstration shows the ability of the system to expand to include new models, parameters and associated thresholds. This is all done with no disruption to the systems condition monitoring capability. If expanded to a deployable system, this pro-

cess of deploying new models may be automated using the information already present in the database, making any such system very flexible and autonomous.

6.7 Conclusion

This chapter has presented a flexible agent architecture utilising diagnostic algorithms to monitor a full gas turbine and its sub-systems. The system utilises both a hidden Markov model-based algorithm to monitor the degradation of each subsystem, and a thresholding algorithm to monitor the individual variables. The Chapter specifies the necessary agents, along with communication protocols and ontology. Together with a custom user interface, these elements fully define a comprehensive gas turbine condition monitoring multi-agent system. A demonstration is provided to illustrate the successful operation of the system, including its ability to absorb new models and information after initial deployment.

Chapter 7

Rule-based Evidence Combination

This chapter provides an approach to combining the several measures and techniques presented in this thesis to provide a rule-based fault diagnosis capability using the information from the existing prototype system. This is possible through the capture of expert knowledge from SSE. The diagnostic process is successfully captured and documented. Several diagnostic rules have been derived and are also presented. The approach is proven through the use of a case study, showing the potential for expansion of the knowledge-base and implementation as a rule-based system. Ultimately, this rule-based system may be incorporated into the multi-agent system presented in Chapter 6 as an agent.

7.1 Evidence Combination

Evidence combination, in the context of fault diagnosis, can be described as an "algorithm to combine all the available evidence into useful and coherent defect information for the engineer" [103]. The combination of evidence is often probabilistic, with Bayesian or belief function approaches as examples of valid methods [154]. However, this chapter proposes the use of expert-derived rules in order to combine raw sensor data, and relevant information from any deployed agents, into coherent diagnostic information.

7.1.1 Reasons for Evidence Combination

As different faults may be indicated with varying levels of success using different algorithms or sets of parameters a comprehensive suite of techniques could be used to provide a more robust detection system. However, with several components, each with several monitored parameters, the number of sources of failure in the system is potentially overwhelming. This is further compounded when multiple algorithms are used. An example [155] summarises seven techniques just for on-line monitoring of generator windings.

All of this data must be interpreted while keeping the data analysis load on the specialist to a minimum. With this in mind it is necessary to perform some automated processing of the data before it reaches the engineers. The system must alert the specialist to potential problems while keeping the number of alarms low. It must provide the specialist with all appropriate information pertaining to the alarm in an efficient way. The outputs from individual agents must then be compared and reconciled with results from other agents, in order to arrive at a fault diagnosis. The requirement to have multiple agents coming to a single conclusion, despite potentially differing local results, requires some form of evidence combination.

7.1.2 Review of Evidence Combination

A MAS has been developed for Iberdrola, called DADICC, an abbreviated name for an "intelligent system for anomaly detection in a combined cycle gas turbine plant" [6]. It uses both anomaly detection and diagnosis agents for each component, before reaching a plant-wide condition appraisal. However, while this allows the components to be considered in isolation, and as a complete system, it uses only a single technique for anomaly detection: ANNs. ANN models are trained for each component, such as the GT compressor or combustion systems, and used to predict outputs, with comparison to real values allowing anomalies to be detected. Diagnosis is performed by a knowledge-based expert system.

The DADICC is a good example of how to divide the plant into its component parts in order to develop an agent architecture (shown in Figure 7.1). However, it requires a expert system for every component. Its reliance on a single anomaly detection technique also means that it does not take advantage of the inherent robustness of corroborating between multiple algorithms.

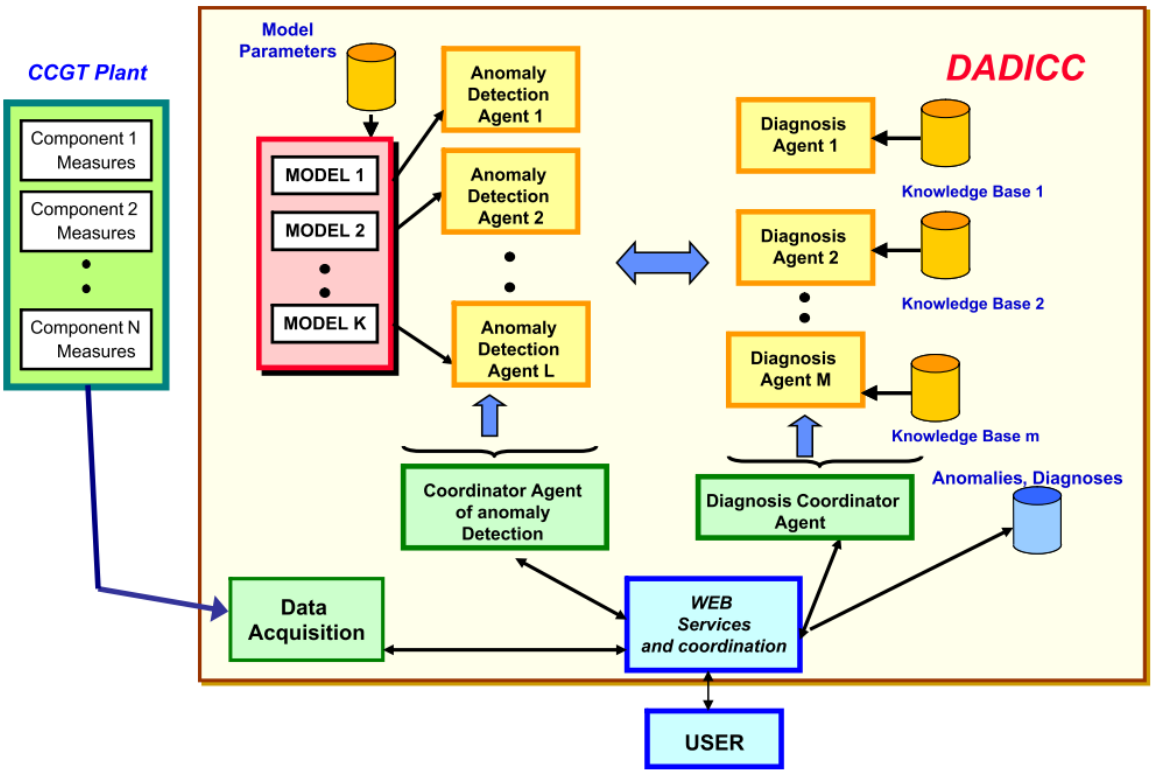

Figure 7.1: The DADICC Architecture

As the type and architecture of plant in question dictates the asset types present, this also influences the algorithms used. A Neural Network Combined Model has been proposed which models all the components of a plant with an ANN [108]. However, this requires knowledge of all the inputs and outputs of the components of the plant. This means that a MBR-style approach may be possible to combine the evidence from each ANN and provide fault diagnosis capability. The example provided used an existing plant model, complete with block diagram and input/outputs, meaning that all this information was already available. While using ANNs removes the need for first-principles models for the components, the process of identifying the plant model and ANN inputs/outputs is still potentially time consuming and will vary between plants. This has not been addressed as the approach has not been applied to an actual plant. Despite the modular approach, it assumes that ANNs are the best choice for all components, where there may be more appropriate algorithms for the different components.

The approach proposed by Sarkar et al. [9] utilises symbolic dynamic filtering (SDF) in order to provide an anomaly measure of the main sub-components of an aero-engine

gas turbine. A high anomaly measure is analogous to a low LL value from the HMMs presented in this thesis, and as such provides a method of fault isolation. However, while this approach may indicate the component that is malfunctioning, it does not provide further information as to the reason for the fault.

An alternative approach to fault diagnosis by Simani [156] uses mathematical models in order to determine the fault and its location, with the use of a ANN layer in order to determine the severity of the fault. The principle disadvantage to this approach is the requirement of detailed mathematical models.

7.1.3 Approach to Evidence Combination

The approach to evidence combination must be carefully considered. Catterson et al. [103] review some techniques for transformer condition monitoring, providing results for the use of several methods. From the results, a Bayesian belief network (BBN) is shown to be the most effective approach overall. It may be possible to link the structure the BBN to the architecture of the prototype MAS system. This would allow the BBN to adapt automatically as new agents and algorithms are added to the system [103].

While the structure of the system would seem to be compatible with the approach described by Catterson et al., this was applied to an application where each agent has already reached a fault diagnosis. As the aim is to utilise evidence combination to provide a fault diagnosis, rather than consolidating several existing diagnosis, this is a distinctly different problem. Even if the approach could be adapted to work in this different context, it would be necessary to have accurate information on the diagnostic performance of more than 100 thresholding agents. This becomes further complicated if the threshold is changed, which would invalidate all of the information.

Other approaches to evidence combination have focused on the Dempster-Shafer theory [157]. This approach was also considered by Catterson et al. but did not perform well. However, the evidential reasoning approach has been developed based on this theory [158], and has seen several applications [159]. These tend to be for condition assessment, rather than fault diagnosis. Successful implementations include an approach utilising an evidential reasoning model alongside a fuzzy model, for power transformer condition assessment [160].

7.1.4 Choice of Approach

While the agents are considered anomaly detection agents, which provide no indication of the cause of the anomaly, it is reasonable to infer that different faults would result in different models and parameters showing anomalies. This would allow the system to provide a fault diagnosis "layer" above the individual anomaly detection agents. An approach to evidence combination that could take advantage of the knowledge present within the utility was more appropriate. This would allow knowledge of the combination of parameters usually linked to particular kinds of faults to provide a fault diagnosis capability as part of the evidence combination approach.

This chapter proposes a form of RBES to consolidate the outputs from the different agents to provide this fault diagnosis functionality. It is proposed that the hidden Markov model and thresholding agents may continue to be used as anomaly detection agents. However, the outputs from these agents, as well as the information provided by the directly observable parameters, may be combined using a rule-base in order to provide a fault diagnosis capability.

7.2 Knowledge Capture

The diagnostic knowledge of the utility is split between two sets of engineers. A set of equipment performance engineers are responsible for monitoring the day-to-day condition of the assets, via a condition monitoring system. There is also a set of asset engineers who are experts on the various items of plant, in this case gas turbines. Knowledge elicitation meetings were arranged with both sets of engineers in order to more effectively capture the knowledge across these two distinct domains.

During the process of elicitation, the CommonKADS framework for knowledge representation was used (Section 2.10.1). The captured knowledge was represented as activity, task and semantic diagrams.

7.2.1 Diagnostic Process

This sub-section presents knowledge of the process engineers follow to diagnose faults. This process, following the raising of an alarm, is shown in an activity diagram in Figure 7.2.

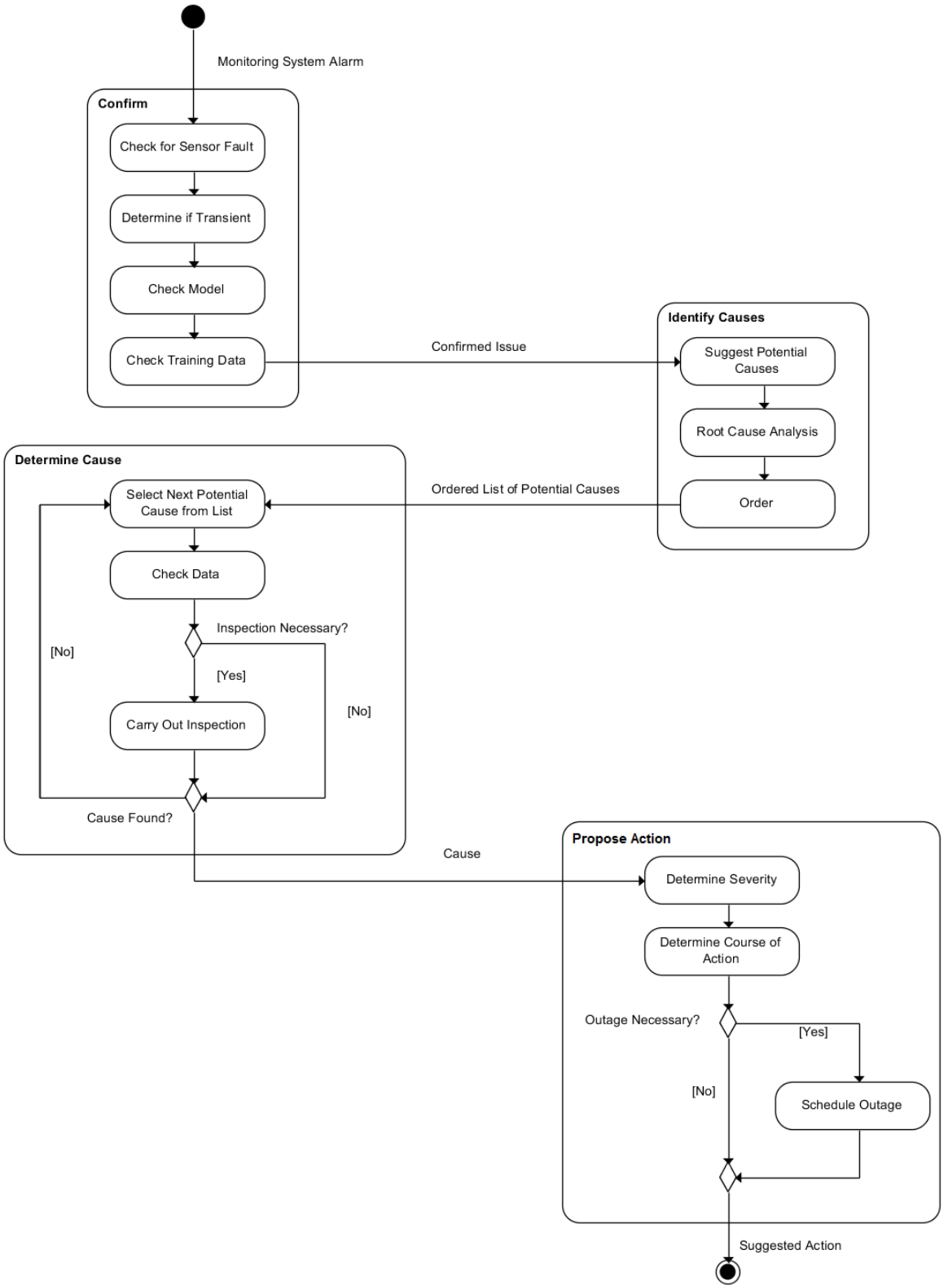

Figure 7.2: Activity Diagram of the utilities process following a monitoring system alarm.

The process begins with an inspection of the data from the condition monitoring system. Equipment monitoring engineers will begin the process of addressing the alarm by establishing if the issue is genuine. The sensor data will be checked to confirm that it is functioning correctly. A malfunctioning sensor may display irrational readings (for example -300°C), may show noisy or unstable readings, or may show readings that are not possible based on other related or closely located sensors. Then the readings will be checked to confirm that the alarm was not caused by a transient. A potential cause of this type of false alarm is a move to a new operating point. If readings return to normal following the change in operating point, then this would suggest it is not a genuine issue.

Following confirmation that the issue is genuine, the model responsible for the alarm (as in Chapter 5 the models are Combustion, Compressor, Fuel Flow, Mechanical and Wheelspace) is checked. It may be possible that another erroneous reading, unrelated to the variable the fault is attributed to, may be affecting the model. Finally, it must be considered whether the model has been trained for the conditions present. It is possible that the ambient conditions may be different to any of the training data. If this is the case then new data will be trained into the model. Following these steps, if the issue is determined to be genuine, and the reason is not immediately obvious, a set of specialist asset engineers take the process to the next step.

The asset engineers are experts on various items of plant. In this case the knowledge was captured from gas turbine specialists. These engineers use their experience to hypothesise on the possible causes of the symptoms displayed by the monitoring system. They will then perform root cause analysis (RCA) to determine the requirements for a hypothesis to be correct, and produce an ordered list of root causes and the necessary checks. This takes into account the likelihood that the potential cause is the real cause, and the impact of the check. If a check requires a long outage for an off-line inspection, or it is unlikely to find the cause, it will be further down the list.

Once the root cause of the issue is determined, the action will be decided on. This will depend on the severity of the fault. For example, if the fault does not prevent the turbine from running at base load, then action may be deferred until a scheduled outage and the operating regime may simply be constrained. Conversely, if the issue is a precursor to a major failure it may be taken off-line immediately. Based on the elicitated knowledge it is possible to derive the task diagram for the process, shown in Figure 7.3.

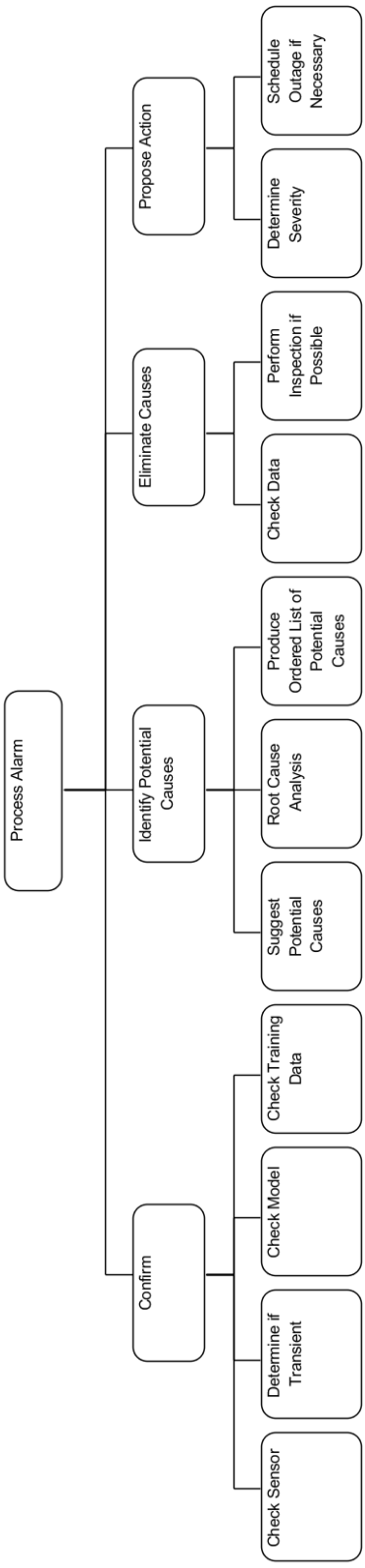

Figure 7.3: Task Diagram for processing alarms.

7.3 Representation of Diagnostic Rules

The following rules can be used when confirming the legitimacy of an issue and determining the likely cause of a fault. They were captured during the knowledge elicitation meetings. They are represented as semantic diagrams.

7.3.1 Loss of Output Due to Ambient Conditions

Loss of output can occur for many reasons. This may be due to faults. However, the domain experts suggested two examples where ambient conditions have caused a loss in output, where the gas turbine is still operating as normal. These are related to the ambient temperature and a change in inlet Differential Pressure (DP) due to the weather conditions, shown in Figure 7.4.

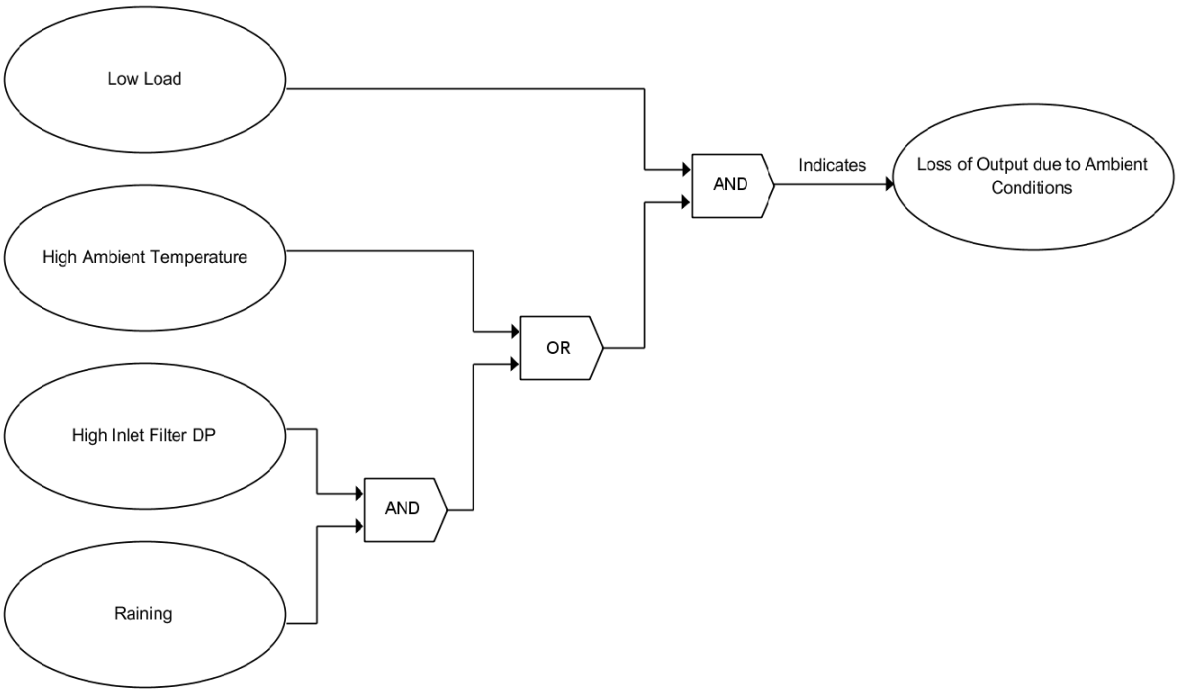

Figure 7.4: Rule #1: Ruling out false alarms due to ambient conditions.

7.3.2 Combustion Cold Spot

A cold spot is an area in the combustion chamber(s) of a gas turbine where temperatures are lower than expected.

A cold spot is usually indicated by low readings on multiple combustion thermocouples. If only on a single Thermocouple (TC) it cannot be a cold spot, as there are multiple thermocouples per combustion can, and is likely a sensor malfunction instead. (This rule may or may not apply depending on the number of thermocouples on the specific turbine).

Figure 7.5 shows that any low temperatures on a single thermocouple may be considered a thermocouple issue. Further elicitation with the domain experts was carried out to find the potential root causes of a cold spot.

Figure 7.5: Rule #2: Check for sensor malfunction

Swirl analysis must also be carried out in order to confirm that the cold spot moves with load. While the prototype condition monitoring system does not have full swirl analysis capability it is possible to approximate it by checking if the cold temperatures move across the thermocouples (they are numbered by location, so this is a feasible implementation option).

Figure 7.6 shows the rule for confirming the presence of a cold spot.

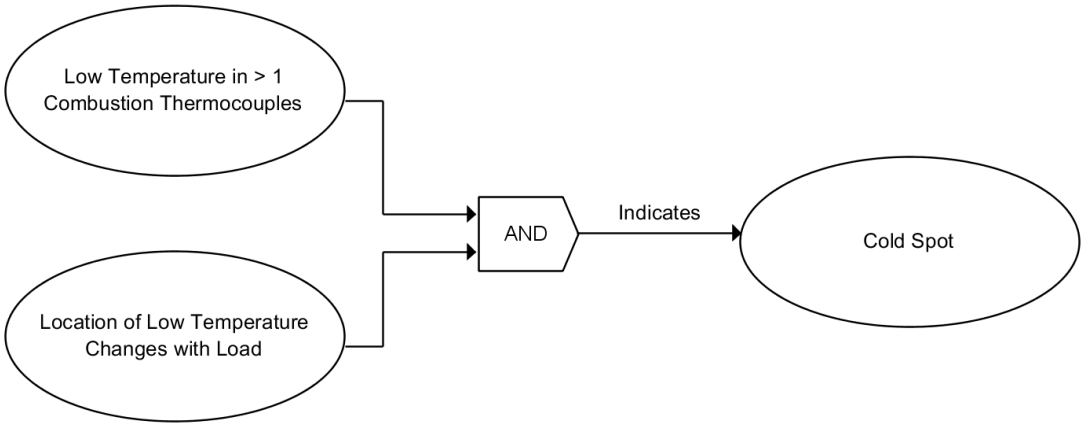

Figure 7.6: Rule #3: Rule for a cold spot

It was found that if a exhaust frame leak occurs, this can lead to cold air entering the turbine, and resulting in a cold spot. It was possible to detect the frame leak through high wheelspace cooling system temperatures. This rule is represented in Figure 7.7.

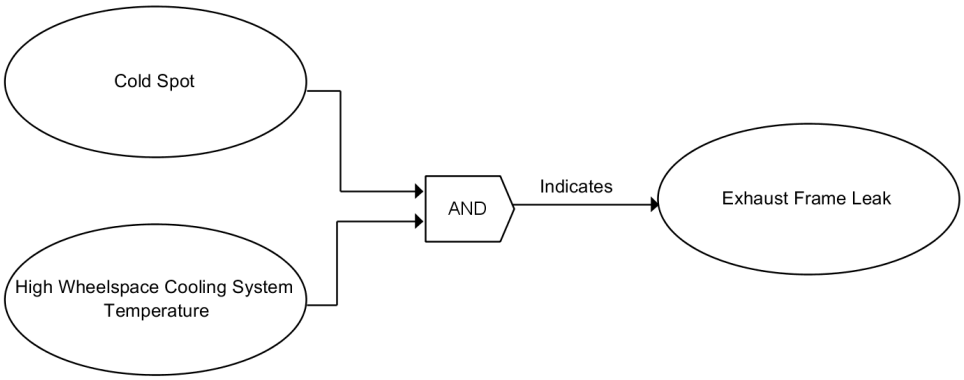

Figure 7.7: Rule #4: Rule for exhaust frame leak

Figure 7.8 shows the conditions necessary for an inspection to be considered necessary. A difference between the cold spot thermocouples and the rest of the thermocouples of more than 20°C is considered serious enough to warrant an inspection. Equally, a smaller difference of 10°C is sufficient when the difference is increasing, as this suggests continuing changes to the GT's behaviour.

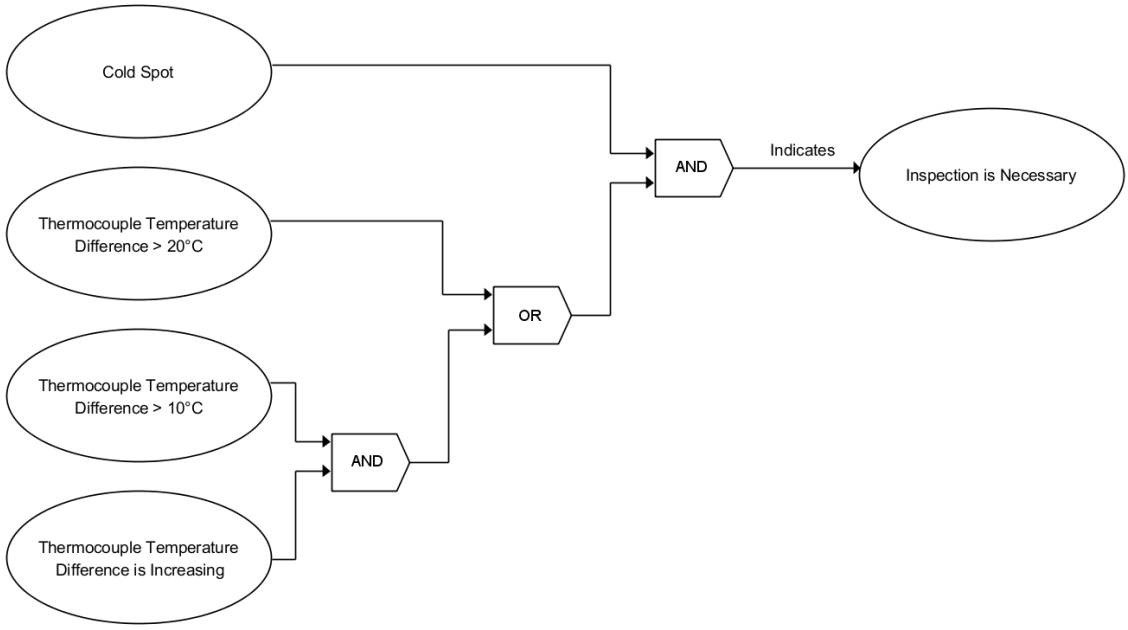

Figure 7.8: Rule #5: Rule for inspection following a diagnosed cold spot

7.3.3 Inlet and Compressor Faults

According to the domain experts, it is possible to detect an IGV motor fault, such as freezing, by comparing two variables: the actual position and the control reference. If they are different this indicates an actuator fault which in turn suggests that the actuator requires resetting. This is shown in Figure 7.9.

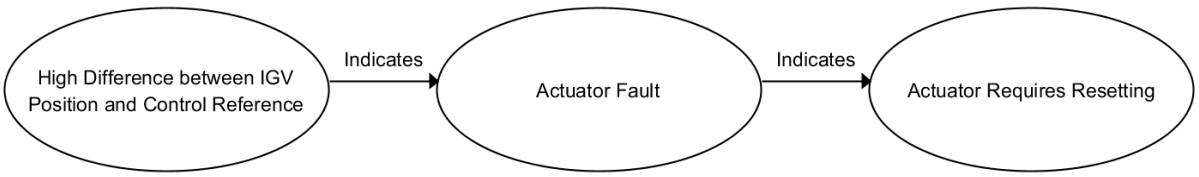

Figure 7.9: Rule #6: Rule for detecting inlet guide vane actuator fault.

If there is high bellmouth differential pressure, which is the pressure difference between the inlet air and that entering the compressor, this suggests that the inlet air filter is blocked. This is shown in Figure 7.10.

Figure 7.11 shows that compressor fouling is detectable by a drop in compressor output temperature and pressure, along with a drop in compressor efficiency and load.

Figure 7.10: Rule #7: Rule for filter blockage

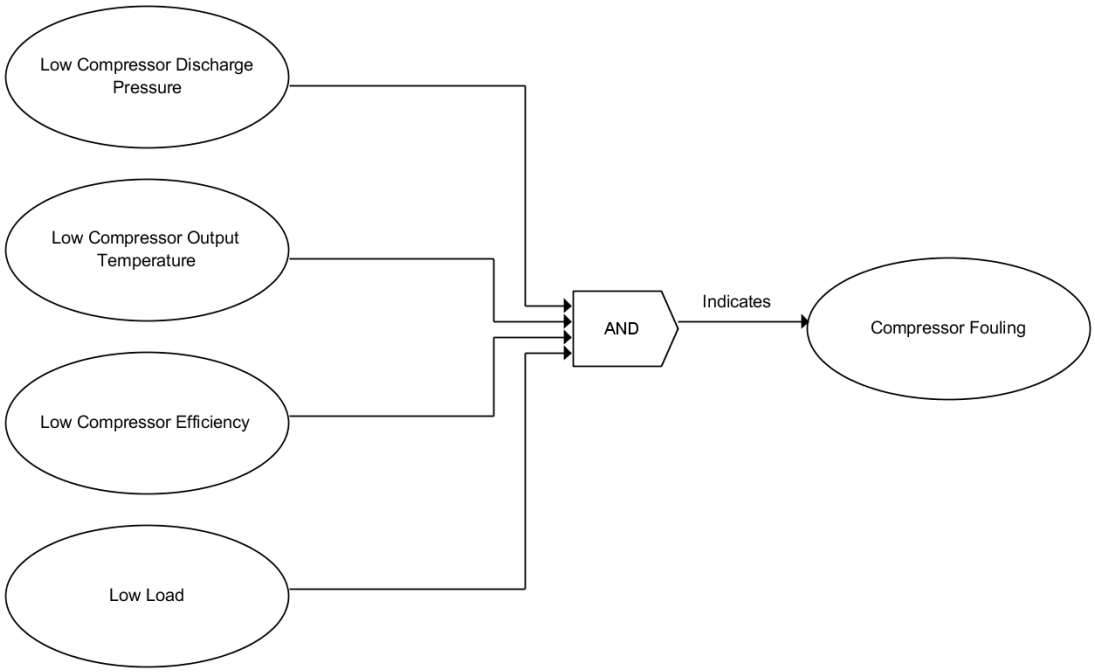

Figure 7.11: Rule #8: Rule for compressor fouling

The domain experts gave another case where there was drop in load, compressor output temperature and pressure, but compressor efficiency was unaffected. This ruled out compressor fouling as a cause. It was also found that fuel flow was reduced and the firing and exhaust temperatures were also reduced. This represented events across three of the five models: compressor, fuel system, and combustion. Upon inspection it was discovered that a compressor blade bleed valve had detached. The semantic diagram in Figure 7.12 represents this knowledge.

Figure 7.12: Rule #9: Compressor Bleed Valve Failure Rule.

7.3.3 a Vibration

It was found that lubrication oil pressure drop could lead to rub between the rotor and the bearings and increase vibration. Lube oil outlet temperatures would also increase. From this captured domain knowledge the rule shown in Figure 7.13 can be inferred.

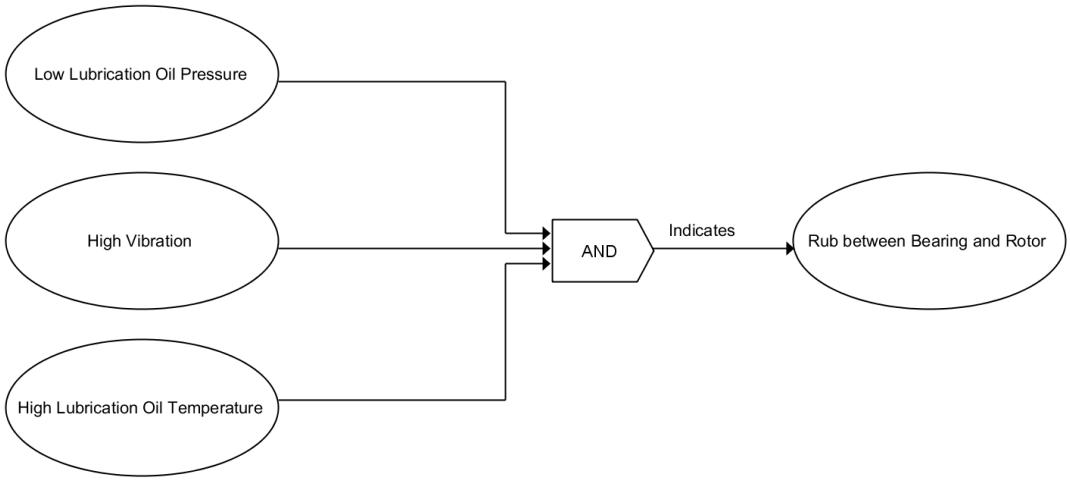

Figure 7.13: Rule #10: Rule for detecting rotor and bearing rub

7.3.4 Combustion Flashback

The domain experts described a set of symptoms that indicated flashback in the combustion system. The Nitrogen Oxides (NOx) level would spike, the fuel valve position move, and the fuel pressure drop. Figure 7.14 shows a rule to diagnose flashback in future in order to prevent damage, such as burner tube failures.

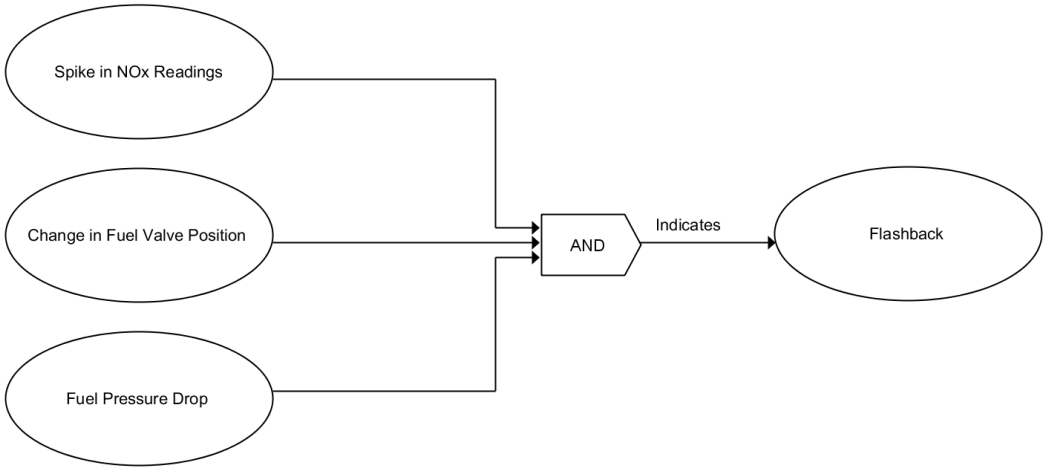

Figure 7.14: Rule #11: Rule for detecting flashback in the combustion system

7.3.5 Fuel System Reverse Flow

Reverse flow is when the flame from the combustors is allowed to propagate back through the compressor. The experts stated that low DP between the compressor discharge air and fuel system can cause this phenomenon. This allowed the rule in Figure 7.15 to be derived. It should be noted, that this rule is only valid for the "find cause" task. The transducer must be confirmed to be functioning correctly as part of the confirmation task, as a drifting transducer could cause this symptom.

Figure 7.15: Rule #12: Rule for detecting reverse flow.

7.3.6 Applying the knowledge to the multi-agent system

For the rule-based evidence combination to begin, one or more of the HMM models must show a change in likelihood sufficient to indicate an anomaly, as indicated in Section 5.8, or a threshold agent must detect a high or low value. The information provided by the agents may be fed into the rule base, such as the affected sub-system (HHM Agents), or the parameters which exceed the high/low threshold (Threshold Agents). Using this and other derived information along with the semantic rules, it may be possible to diagnose the issue.

7.4 Case Study - Exhaust Frame Leak

The following case study demonstrates the application of rule-based evidence combination to anomalies relating to an exhaust frame leak.

A new set of data was taken from a different time period than that which was used previously (Chapter 5), consisting of a complete year's worth of data from two GTs. No information was given as to the condition of the turbines over the time period, other than the existence of an exhaust frame leak-related issue during month 6. However, which GT this applied to was not stated. This was in order to provide a useful blind test

of the MHMMs. However, because of this it was difficult to determine an appropriate training set, as it was not possible to determine at what times the machine was operating normally. Under the assumption that the machine is operating without faults the majority of the time, the solution was to use the entire year as a training set. While this means that the anomalous behaviour is also trained in, reducing the expected drop in LL, it is still far rarer than normal conditions, and would still result in a drop in LL.

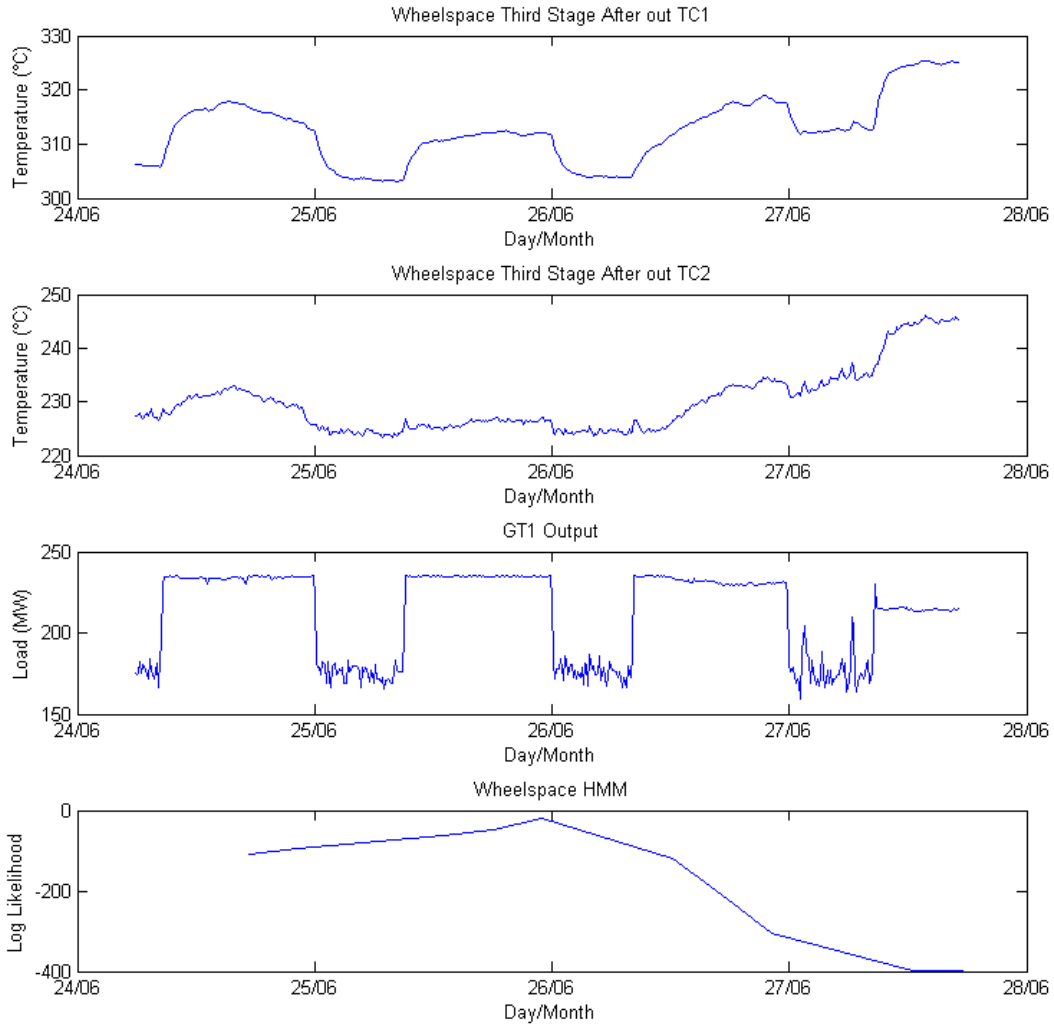

Figure 7.16: Rise in wheelspace temperatures as the LL decreases.

Upon inspection, it was found that the wheelspace model of GT1 showed a decrease

on the 27th of month 6, just before an outage. Figure 7.16 shows that around the same time the third stage outer wheelspace temperatures increase. While these values are somewhat proportional to load, the increase is clearly disproportionate to those that came before.

The fact that the increase is shown on both thermocouples suggests that this is not caused by a sensor issue. The issue cannot be dismissed as a transient, as it persists and increases through and after a change in load. No other variables were found to be feeding erroneous values to the model, and the data was included in the training set, so it cannot be that the model has not seen the data before. These checks fulfil the requirements of the "Confirm" task (See Figure 7.3). This allows us to continue to the "identify potential causes" and "eliminate potential causes" tasks, which are represented by the rules defined in section 7.3.

There is only one rule that includes high wheelspace temperatures, Rule #4, shown in Figure 7.7, but this requires that there is a cold spot present. Therefore we are unable to attribute a cause to this anomaly at this time.

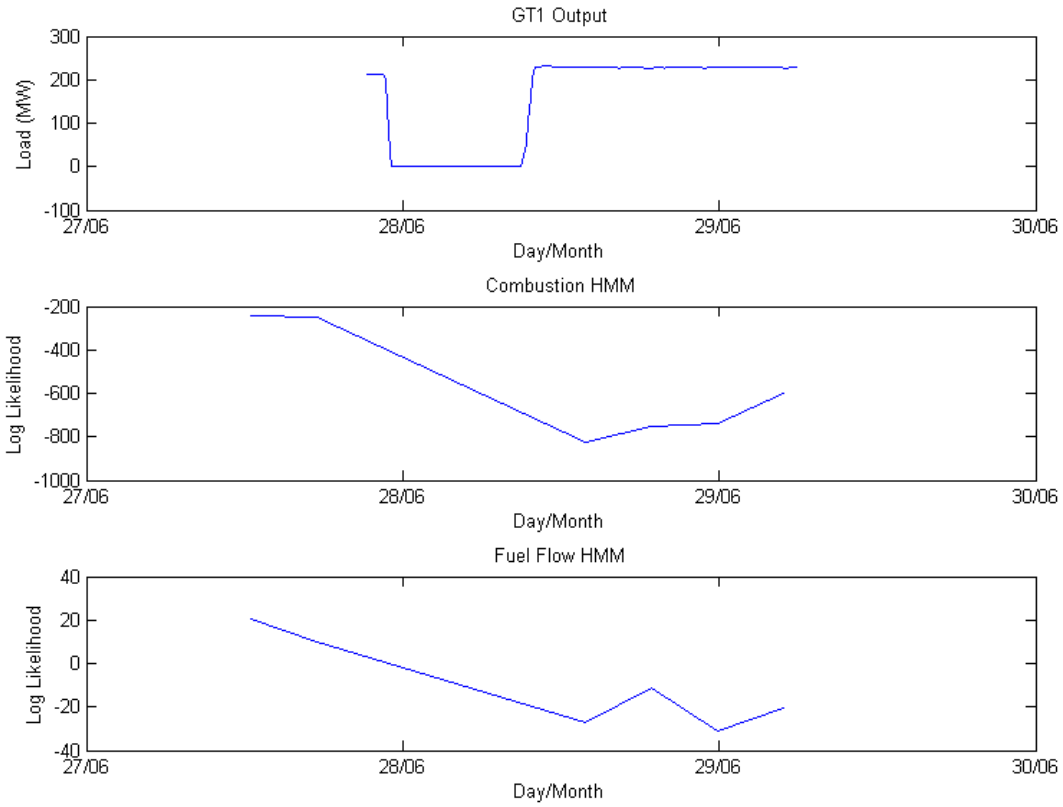

Figure 7.17: Drop in LL following outage

Following the outage, the combustion and fuel flow model shows a drop in LL (Figure 7.17). At the same time, the temperature readings of several TCs decrease, while others do not. As more than one thermocouple shows low temperatures, Rule #2 (Figure 7.5) allows us to determine that this is not a thermocouple malfunction. The gas turbine has 31 TCs, arranged around the combustion chamber, where #31 is adjacent to #1. The thermocouple readings are shown in Figure 7.18, where TCs #28-31 show a drop in temperature, while #26-27 seem unaffected. While the temperature across the combustion chamber can be influenced by many effects, including load, an isolated drop suggests a cold spot.

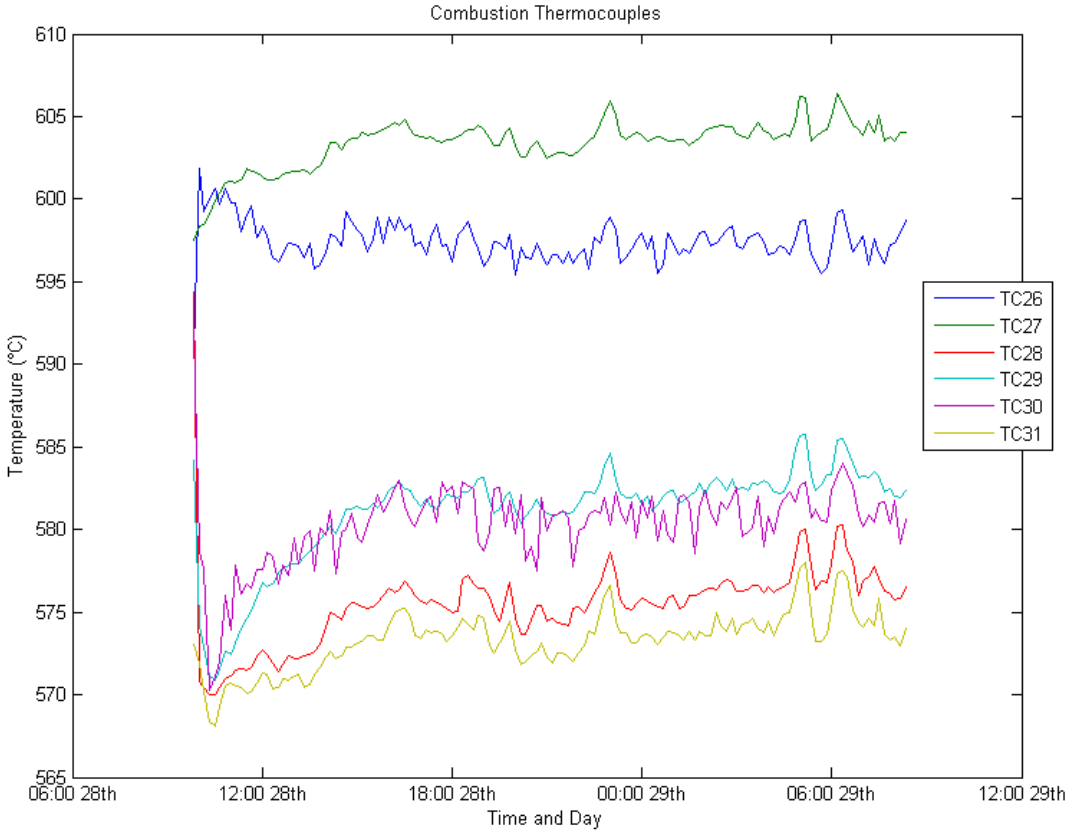

Figure 7.18: Drop in multiple thermocouples following outage

After reaching this conclusion, discussion with the asset engineers confirmed that the diagnosis of a cold spot at this time was correct. The asset engineers explained that following this conclusion, they performed swirl analysis by dropping the load twice. This is necessary to fulfil Rule #3 (Figure 7.6), which requires that the position of the temperature drop must change or "swirl" with load. A drop to 200 Watts is shown in Figure 7.19).

Along with the first load drop (Figure 7.19), TCs #28-30, which were previously low, show the temperatures increase. At the same time, TC #31 shows a further drop, and #1 now decreases also. This suggests that the cold spot has moved along with the change in load.

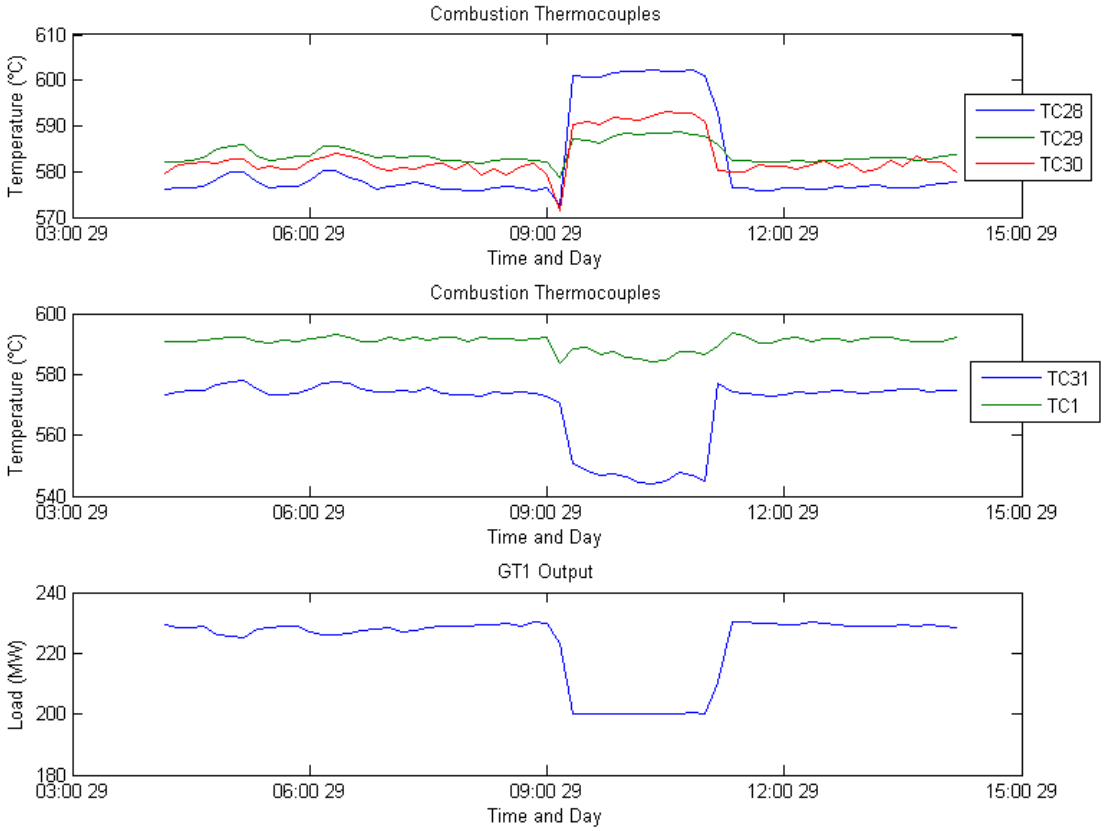

Figure 7.19: Change in thermocouples following load drop to 200MW.

The second load drop (Figure 7.20) shows an increase in TC #32, at the same time as TC #1 decreases further. This again shows that the cold spot is moving around the turbine dependant on load. With this "swirl" established, the requirements for the Rule #3 (Figure 7.6) are now met.

With a cold spot confirmed, this can now propagate through the rule-base, and establish the presence of an exhaust frame leak based on Rule #4 from Figure 7.7.

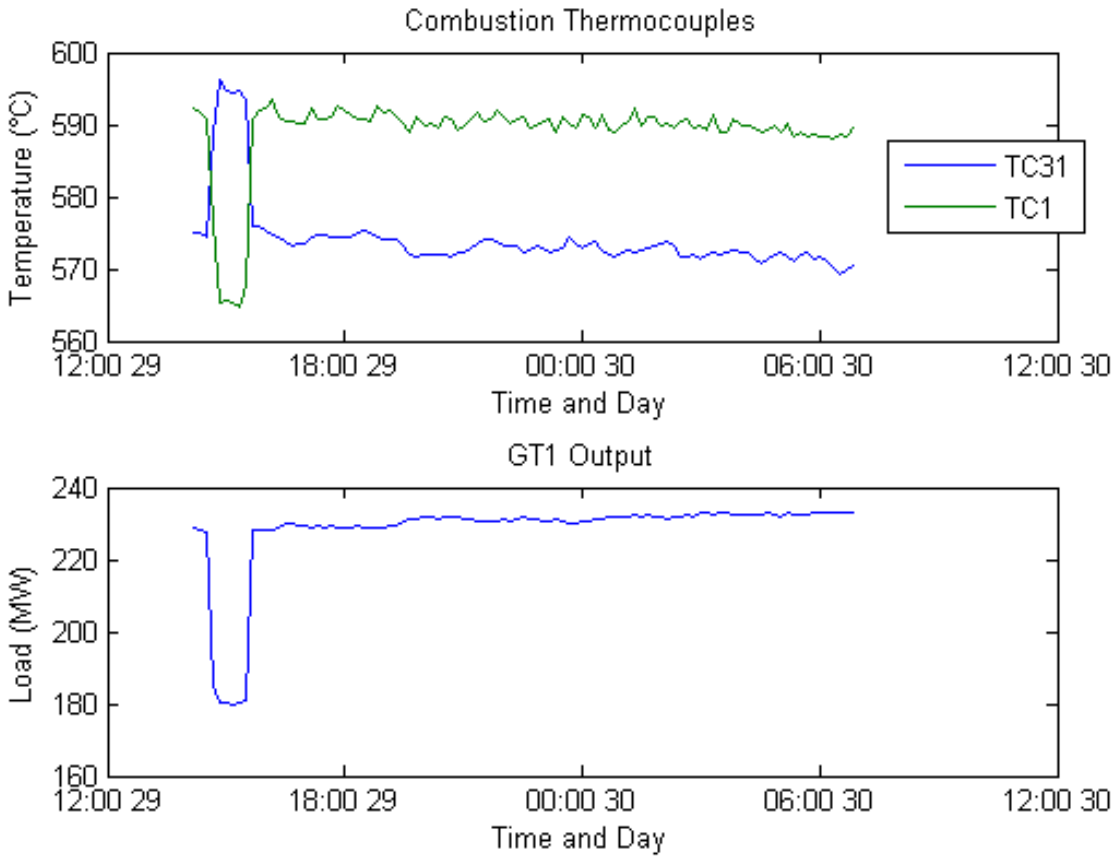

Figure 7.20: Change in thermocouples following load drop to 180MW.

7.5 Conclusion

This chapter has shown the successful capture of expert knowledge, to allow the diagnosis of faults in gas turbines. This rule-based evidence combination approach utilises the output of the diagnostic algorithms implemented by the MAS in Chapter 6. It has been proven through a case study utilising real-world data collected from an operational gas turbine. The HMM algorithm was able to detect the presence of an anomaly, and the proposed rule-based approach to evidence combination was able to diagnose the fault. This approach may be extended an deployed as an agent, and incorporated into the condition monitoring MAS.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

The monitoring of major and critical items of a power plant can allow prevention of a major outage, a longer operating lifetime and a reduction in the cost over that lifetime. Artificial intelligence techniques may be applied to this field to provide enhanced condition monitoring. This thesis has outlined several intelligent techniques, and a data mining methodology in order to choose and configure the most appropriate. Several of the techniques are applied, tested and evaluated.

The application of artificial neural networks to a ball and race-type pulverising mill was shown to be capable of detecting faults in the primary air system before existing methods. Furthermore, several techniques for monitoring gas turbines were trialled, but an algorithm based on hidden Markov models using Gaussian mixture models proved most effective. These models were capable of detecting faults in a gas turbine across five primary subsystems, despite a large sampling period. The HMMs were compared to an existing commercial and industrially implemented condition monitoring system. The results showed a capability to successfully detect faults not possible with the existing system, and generally higher performance.

A detailed analysis of techniques for automatically interpreting the output from the HMMs using attributes of the log-likelihood plots are provided. This includes new approaches considering the rate of change and sequence variance shown to be capable of detecting anomalies. The application of linear regression is shown to allow the trending of degradation in the turbine.

A single technique may be appropriate for a single item. But it is not uncommon for

multiple techniques to be used across a plant, or even within the same asset. Multi-agent Systems provide a convenient and extensible way to integrate these techniques. However, multi-agent and/or multi-technique condition monitoring approaches are not necessarily better than simpler systems. There must be a mechanism to allow the results from multiple agents to be reconciled and provide the user with not just more information, but the most accurate, useful and relevant information.

This thesis has provided a review of multi-agent system technology in the power generation domain. A set of established standards are outlined and proposed for the implementation of the selected algorithms. A prototype is presented, implementing the HMMs as agents. Alongside these HMM agents, a set of thresholding agents are provided in order to give parameter-specific monitoring capabilities. The system is implemented using the full FIPA standards.

A rule-based approach to evidence combination is designed, and the process of knowledge capture is outlined, and the derived rules presented. Finally, a case study is provided showing the successful detection of an anomaly by the HMMs, and the use of the captured knowledge, in the form of rules, to allow diagnosis of the issue.

This thesis has presented condition monitoring algorithms that can capture the relationship of complex non-linear systems and, in the case of MHMMs, capture the dynamics of the system despite low frequency data. The outlined agent-based architecture allows several algorithms to work together to provide fault detection, isolation, and diagnostic capability. These technologies can be used to improve the condition monitoring of large items of plants.

8.2 Further Work

The following areas have been identified as areas for further research.

8.2.1 Improved Artificial Neural Networks for Pulverising Mills

The ANN presented in Chapter 4 was tested primarily on a single state. ANNs could be trained for the other plant states. This would also require a second mechanism for identifying plant state. The approach using K-means in Section 4.6 already shows this ability. Further work should focus on developing this mechanism and training ANNs for the full range of mill operation.

The technique would also be applicable to the feeder and temperature variables, and this would be necessary to provide a full anomaly detection suite for the pulverising mill. This may be possible through a generic agent, where only the input variables and the weights file change. This would show the re-usability of the condition monitoring agents.

Possibilities for improving the technique include experimenting with ANN settings, and other ANN types and architectures. The use of delayed time values (t-1, t-2 etc.) as input could potentially improve the accuracy of predictions, allowing the network to learn the behaviour of variables over time, as opposed to simply the relative relationships between them at any point in time as is the case with the current approach. This was shown to be the case and was subsequently utilised in the ANNs for gas turbines (Chapter 5). This may also capture the changes in state that currently pose an obstacle to the use of a single ANN for all states.

It is also possible that other measures of ANN accuracy may provide a more "engineer friendly" anomaly detection system [17]. While the Pearson correlation coefficient shows promise, it is a rather abstract measure. As an alternative or in addition to this it may be useful to view the raw residuals between normal and predicted values, which can either be interpreted by an engineer, or other software solutions.

8.2.2 Improved Hidden Markov Models for Gas Turbines

While the HMMs presented in this thesis have been shown to be capable of condition monitoring of gas turbines, there is scope to improve their performance. Most of this is possible by tailoring the properties of the model and the training parameters to the particular data set of each of the five sub-systems models, rather than using a general model architecture and training algorithm. For example, the same size of model (number of states and mixtures) was used for all sub-systems. A different size of model as appropriate to each sub-system may increase the accuracy of the models [161].

During training, some of the models' observation probability distributions were found to tend towards a point function. This may be addressed by adjusting the prior added to the covariance matrix during training [148]. The algorithm as implemented does not permit adjustment of this value without direct manipulation of the code. Future versions should address this by allowing this to be adjusted as an option, or retraining automatically with an adjusted prior when necessary.

The initialisation of a GMM is especially important, and the EM algorithm only finds

a local maximum [48]. The implementation uses random values for initialisation. A more sophisticated approach using a statistical analysis of the data may allow more appropriate initial values and thus improve the results of the algorithm.

8.2.3 Improved Threshold Agents

While the thresholding agents provide a useful indication of anomalous conditions with respect to a single variable, there are somewhat limited in capability. While a parameter exceeding the limits experienced during the training period is often an indicator of an anomaly, it is possible that the parameter may be showing previously unseen values because the ambient or operational conditions may be different from what it was trained on. A method to allow the presence of previously unseen conditions to be taken into consideration before raising an alarm may result in less false positives, resulting in a more effective approach.

As several variables are dependent on operational and ambient conditions, the anomaly detection algorithm could go further. For example, many of the parameters are dependent on load. It may be possible to train several thresholds into the agent, each for different operating levels.

8.2.4 Further Development of Rule-based Evidence Combination

Examples of the use of rule-based evidence combination have been provided in this thesis. However, while the rule base has been established, it has not been implemented as a software system. Future work should focus on development of such a system and its addition as an agent to the existing multi-agent system. This would also require the inclusion of the automated interpretation techniques proposed for the HMM results, and the improved thresholding agents.

The rule base should also be expanded. Further knowledge elicitation interviews with utility staff and experts should be arranged to allow the rule base to be extended to provide as comprehensive a fault diagnostic capability as possible.

8.2.5 Additional Algorithms

This thesis presents several techniques from the field of artificial intelligence. While success has been found with ANNs and HMMs, several other algorithms are reviewed.

It may be that other techniques may be applicable to gas turbine condition monitoring. With the use of a multi-agent architecture for the prototype, it would be relatively easy to implement one or more of these approaches as an agent.

Combining the use of multiple algorithms would increase the robustness of the system. However, this is only possible if the algorithms can be combined in an efficient way, using appropriate forms of evidence combination. Rule based evidence combination has already been applied successfully for fault diagnosis. This may be extended to take into account the results from other algorithms, or other forms of evidence combination may be necessary before reaching the rule-base's level.

8.2.6 Application to other Rotating Machinery

While the HMM-based approach to anomaly detection using low-frequency data has been shown using data from the combustion turbines of a CCGT, the technique could be applied to other applications. This would include other rotating plant items, such as steam turbines, or wind turbine drive trains.

Recent work has applied the HMM-based anomaly detection technique to wind turbine SCADA data, taken at 10-minute sampling intervals, and shows promising results. However, further investigation would be required for a full application.

References

[1] P. J. Tavner, "Review of condition monitoring of rotating electrical machines," *IET Electric Power Applications*, vol. 2, no. 4, pp. 215 – 247, 2008. (cited on pages 1, 83)

[2] M. Markou and S. Singh, "Novelty detection: a review part 1: statistical approaches," *Signal Processing*, vol. 83, pp. 2481–2497, Dec. 2003. (cited on page 1)

[3] Greg Stanley and Associates, "A Guide to Fault Detection and Diagnosis," *White Papers for Operations Management Automation. Available online at* `http://gregstanleyandassociates.com/whitepapers/FaultDiagnosis/faultdiagnosis.htm`*; accessed 20 December 2012.*, 2010. (cited on page 1)

[4] A. Heng, S. Zhang, a. Tan, and J. Mathew, "Rotating machinery prognostics: State of the art, challenges and opportunities," *Mechanical Systems and Signal Processing*, vol. 23, pp. 724–739, Apr. 2009. (cited on page 2)

[5] A. J. Brown, V. M. Catterson, M. Fox, D. Long, and S. D. J. McArthur, "Learning Models of Plant Behavior for Anomaly Detection and Condition Monitoring," *2007 International Conference on Intelligent Systems Applications to Power Systems*, pp. 1–6, Nov. 2007. (cited on pages 2, 52, 54, 84, 102)

[6] A. Arranz, A. Cruz, M. Sanzbobi, P. Ruiz, and J. Coutino, "DADICC: Intelligent system for anomaly detection in a combined cycle gas turbine plant," *Expert Systems with Applications*, vol. 34, pp. 2267–2277, May 2008. (cited on pages 2, 84, 151)

[7] K. Jordal, "Variations in Gas-Turbine Blade Life and Cost due to Compressor Fouling A Thermoeconomic Approach," *Applied Thermodynamics*, vol. 5, no. 1, pp. 37–47, 2002. (cited on pages 2, 80)

[8] K. Mathioudakis, A. Stamatis, A. Tsalavoutas, and N. Aretakis, "Performance analysis of industrial gas turbines for engine condition monitoring," *Proceedings of the Institution of Mechanical Engineers, Part A: Journal of Power and Energy*, vol. 215, pp. 173–184, Jan. 2001. (cited on page 2)

[9] S. Sarkar, K. Mukherjee, A. Ray, and M. Yasar, "Fault Diagnosis and Isolation in Aircraft Gas Turbine Engines," in *2008 American Control Conference*, pp. 2166–2171, 2008. (cited on pages 2, 80, 152)

[10] Y. Zhang, Q. Wu, J. Wang, G. Oluwande, D. Matts, and X. Zhou, "Coal mill modeling by machine learning based on onsite measurements," *IEEE Transactions on Energy Conversion*, vol. 17, pp. 549–555, Dec. 2002. (cited on pages 4, 56, 57, 58, 59, 67)

[11] X.-F. Li, Y.-X. Zeng, J. Sun, Y. Li, and H.-Y. Wu, "Fuzzy Optimization Control System and its Application in Ball Mill Pulverizing System," in *2006 IEEE International Conference on Fuzzy Systems*, pp. 615–620, 2006. (cited on pages 4, 56, 59, 60, 67)

[12] S. Chen, S. A. Billings, and P. M. Grant, "Non-linear system identification using neural networks," *International Journal of Control*, vol. 51, pp. 1191–1214, Jan. 1990. (cited on pages 4, 16)

[13] Turbine Services Limited, "Tiger: Summary of Functionality," *available online at* `http://www.turbineserviceslimited.com/Tiger/A4%20Format%20Docs/Summary%20of%20Functionality/Summary%20of%20Functionality%20-%20uk.pdf`; *accessed 20 December 2012.* (cited on pages 5, 86)

[14] A. D. Kenyon, V. M. Catterson, S. D. J. McArthur, and J. Twiddle, "Application of Hidden Markov Models to Gas Turbine Anomaly Detection," *Accepted, IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*.

[15] A. D. Kenyon, V. M. Catterson, and S. D. J. McArthur, "Development of an Intelligent System for Detection of Exhaust Gas Temperature Anomalies in Gas Turbines," *Insight: The Journal of the British Institute of Non-Destructive Testing*, vol. 52, no. 8, 2010. (cited on page 99)

[16] A. D. Kenyon, V. M. Catterson, and S. D. J. McArthur, "Development of an Intelligent System for Detection of Exhaust Gas Temperature Anomalies in Gas Turbines," *CM 2010 and MFPT 2010 : The Seventh International Conference on Condition Monitoring and Machinery Failure Prevention Technologies*, vol. 52, no. 8, pp. 419–423, 2010.

[17] A. D. Kenyon, V. M. Catterson, S. D. J. McArthur, and J. Twiddle, "Design of an Intelligent Diagnostic Architecture to Support the Condition Monitoring of Power Generation Assets," in *Proceedings of the 44th International Universities Power Engineering Conference (UPEC)*, 2009. (cited on page 174)

[18] G. P.-S. U. M. Fayyad and P. Smith, *From Data Mining to Knowledge Discovery: An Overview.* Cambridge, MA: MIT Press, 1996. (cited on page 8)

[19] G. Wang, J. Hu, Q. Zhang, X. Liu, and J. Zhou, "Granular computing based data mining in the views of rough set and fuzzy set," in *2008 IEEE International Conference on Granular Computing*, no. 60573068, p. 67, Aug. 2008. (cited on page 8)

[20] J. A. Steele, *A Methodology for the Analysis of Data within Safety Related and Mission Critical Environments.* PhD thesis, University of Strathclyde, 2003. (cited on page 8)

[21] J. de Kleer and B. Williams, "Diagnosing multiple faults," *Artificial Intelligence*, vol. 32, pp. 97–130, Apr. 1987. (cited on page 11)

[22] K. Pearson, "On lines and planes of closest fit to systems of points in space," *Philosophical Magazine*, vol. 2, no. 6, pp. 559–572, 1901. (cited on pages 11, 13)

[23] I. T. Jolliffe, *Principal Component Analysis*. second ed., 2002. (cited on page 11)

[24] L. I. Smith, "A tutorial on Principal Components Analysis," *available online at* `http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf`; *accessed 20 December 2012.* (cited on page 11)

[25] T.-W. Lee, *Independent component analysis: Theory and applications*. Springer, 1st ed., 1998. (cited on page 12)

[26] A. Hyvärinen and E. Oja, "Independent component analysis: algorithms and applications.," *Neural networks : the official journal of the International Neural Network Society*, vol. 13, no. 4-5, pp. 411–30, 2000. (cited on pages 12, 13, 14)

[27] P. Comon, "Independent component analysis, A new concept?," *Signal Processing*, vol. 36, pp. 287–314, Apr. 1994. (cited on page 12)

[28] L. Zhonghai, Z. Yan, J. Liying, and Q. Xiaoguang, "Application of Independent Component Analysis to the aero-engine fault diagnosis," in *Chinese Control and Decision Conference*, no. 4, pp. 5330–5333, June 2009. (cited on page 13)

[29] W. S. McCulloch and W. H. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bulletin of Mathematical Biophysics*, vol. 5, pp. 115–133, 1943. (cited on pages 14, 16)

[30] C. J. C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery*, vol. 2, pp. 121–167, 1998. (cited on page 14)

[31] "The WEKA Toolkit.," *Available online: http://www.cs.waikato.ac.nz/ml/weka/*. (cited on page 14)

[32] D. Pelleg and A. Moore, "X-means: Extending K-means with Efficient Estimation of the Number of Clusters," in *Proceedings of the Seventeenth International Conference on Machine Learning.*, pp. 727 – 734, 2000. (cited on pages 14, 15)

[33] J. B. MacQueen, "Some Methods for classification and Analysis of Multivariate Observations," in *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281–297, 1967. (cited on page 15)

[34] M. Ester, H.-p. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," in *Proc. 2nd International Conference on Knowledge Discovery and Data Mining*, pp. 226–231, 1996. (cited on page 15)

[35] S. Haykin, *Neural Networks - A Comprehensive Foundation*. Prentice Hall, 2nd ed., 1999. (cited on page 16)

[36] K.-I. Funahashi, "On the Approximate Realization of Continuous Mappings by Neural Networks," *Neural Networks*, vol. 2, pp. 183–192, 1989. (cited on page 16)

[37] S. Russell and P. Norvig, *Artificial Intelligence A Modern Approach*. Prentice Hall, 1st ed., 1995. (cited on pages 16, 17, 39)

[38] T. Kohonen, "The Self-Organizing Map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464 – 1480, 1990. (cited on page 18)

[39] L. Tarassenko, *A guide to neural computing applications*. Butterworth-Heinemann, 1998. (cited on page 18)

[40] J. Vesanto, J. Himberg, E. Alhoniemi, and J. Parhankangas, "Self-organizing map in Matlab : the SOM Toolbox," in *Proceedings of the Matlab DSP Conference*, pp. 35–40, 1999. (cited on page 18)

[41] D. L. Poole and A. K. Mackworth, *Artificial Intelligence: Foundations of Computational Agents*. Cambridge University Press, 2010. (cited on page 20)

[42] V. Arkov, G. Kulikov, T. Breikin, and V. Patel, "Chaotic and Stochastic Processes : Markolv Modeling Approach," in *1st International Conference, Control of Oscillations and Chaos*, no. 5, pp. 488–491, 1997. (cited on pages 21, 85)

[43] A. Poncet, T. P. V. Hoff, K. S. Stadler, and T. P. von Hoff, "Probabilistic Diagnosis of Thermal Plants Condition," in *9th International Conference on Probabilistic Methods Applied to Power Systems*, pp. 1–6, June 2006. (cited on pages 21, 83)

[44] I. Morgan, S. Member, H. Liu, and S. Member, "Predicting Future States With n - Dimensional Markov Chains for Fault Diagnosis," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 5, pp. 1774–1781, 2009. (cited on page 21)

[45] L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989. (cited on pages 21, 25, 26, 30)

[46] L. E. Baum and J. A. Eagon, "An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology," *Bulletin of the American Mathematical Society*, vol. 73, pp. 360–364, May 1967. (cited on page 25)

[47] L. E. Baum and G. R. Sell, "Growth Transformations For Functions On Manifolds," *Pacific Journal of Mathematics*, vol. 27, no. 2, pp. 211–227, 1968. (cited on page 25)

[48] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society Series B Methodological*, vol. 39, no. 1, pp. 1–38, 1977. (cited on pages 26, 175)

[49] F. Dellaert, "The Expectation Maximization Algorithm," *IEEE Signal Processing Magazine*, vol. 13, no. 6, pp. 1–7, 2002. (cited on page 26)

[50] P. Baldi and Y. Chauvin, "Smooth On-Line Learning Algorithms for Hidden Markov Models," *Neural Computing*, vol. 6, no. 2, pp. 305–316, 1994. (cited on page 26)

[51] P. G. Bagos, T. D. Liakopoulos, and S. J. Hamodrakas, "Faster Gradient Descent Training of Hidden Markov Models , Using Individual Learning Rate Adaptation," in *Grammatical Inference: Algorithms and Applications* (G. Paliouras and Y. Sakakibara, eds.), Lecture Notes in Computer Science, pp. 40–52, Springer Berlin / Heidelberg, 2004. (cited on page 26)

[52] L. E. Baum, T. E. D. Petrie, G. Soules, N. Weiss, and B. Y. L. E. Baum, "A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains," *The Annals of Mathematical Statistics*, vol. 41, no. 1, pp. 164–171, 1970. (cited on page 26)

[53] L. R. Welch, "Hidden Markov Models and the Baum-Welch Algorithm," *IEEE Information Theory Society Newsletter*, vol. 53, no. 4, pp. 1, 10–13, 2003. (cited on pages 26, 101)

[54] D. A. Reynolds and R. C. Rose, "Robust Text-Independent Speaker Identification Using Gaussian Mixture Speaker Models," *IEEE Transactions on Speech and Audio Processing*, vol. 3, no. 1, pp. 72–83, 1995. (cited on page 28)

[55] G. D. Forney, "The Viterbi Algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973. (cited on page 29)

[56] M. Fox, M. Ghallab, G. Infantes, and D. Long, "Robot introspection through learned hidden Markov models," *Artificial Intelligence*, vol. 170, pp. 59–113, Feb. 2006. (cited on pages 30, 84, 142)

[57] J. Aldrich, "Doing Least Squares: Perspectives from Gauss and Yule," *International Statistical Review / Revue Internationale de Statistique*, vol. 66, p. 61, Apr. 1998. (cited on page 31)

[58] J. F. Kenney and E. S. Keeping, "Linear Regression and Correlation," in *Mathematics of Statistics, Part 1*, ch. 15, pp. 252–285, Van Nostrand, 3rd ed., 1962. (cited on page 32)

[59] P. Berka, "NEST: A Compositional Approach to Rule-Based and Case-Based Reasoning," *Advances in Artificial Intelligence*, vol. 2011, pp. 1–15, 2011. (cited on page 34)

[60] C. R. Marling, G. J. Petot, and L. S. Sterling, "INTEGRATING CASE-BASED AND RULE-BASED REASONING TO MEET MULTIPLE DESIGN CONSTRAINTS," *Computational Intelligence*, vol. 15, no. 3, pp. 308–332, 1999. (cited on page 34)

[61] G. F. Luger, *Artificial intelligence: Structures and strategies for complex problem solving*. Addison Wesley Longman, 3rd ed., 1998. (cited on pages 34, 40)

[62] I. Watson and F. Marir, "Case-based reasoning : A review," *The Knowledge Engineering Review*, vol. 9, no. 4, pp. 327–354, 1994. (cited on page 34)

[63] G. Schreiber, H. Akkermans, A. Anjewierden, R. de Hoog, N. Shadbolt, W. V. de Velde, and B. Wielinga, *Knowledge Engineering and Management: The CommonKADS Methodology*. MIT Press, 1999. (cited on pages 34, 36)

[64] R. Studer, V. R. Benjamins, and D. Fensel, "Knowledge Engineering : Principles and methods," *Data & Knowledge Engineering*, vol. 25, no. 1-2, pp. 161–197, 1998. (cited on pages 34, 35)

[65] M. L. G. Shaw and B. R. Gaines, "The Synthesis of Knowledge Engineering and Software Engineering," *Advanced Information Systems Engineering, Lecture Notes in Computer Science*, vol. 593, pp. 208–220, 1992. (cited on page 34)

[66] J. Angele, D. Fensel, D. Landes, and R. Studer, "Developing Knowledge-Based Systems with MIKE," *Automated Software Engineering*, vol. 418, pp. 389–418, 1998. (cited on page 34)

[67] H. Eriksson, Y. Shahar, S. W. Tu, A. R. Puerta, and M. A. Musen, "Artificial Intelligence Task modeling with reusable problem-solving methods," *Science*, vol. 79, no. 94, pp. 293–326, 1995. (cited on page 34)

[68] S. E. Rudd, *Knowledge-Based Analysis of Partial Discharge Data*. PhD thesis, University of Strathclyde, 2010. (cited on page 34)

[69] G. Schreiber, B. Wielinga, and J. Breuker, *KADS: A Principled Approach to Knowledge-Based System Development*. Academic Press, 1st ed., 1993. (cited on page 34)

[70] J. K. C. Kingston, J. I. M. G. Doheny, and I. A. N. M. Filby, "Evaluation of workbenches which support the CommonKADS methodology," *The Knowledge Engineering Review*, vol. 10, no. 3, pp. 269–300, 1995. (cited on page 34)

[71] G. Schreiber, B. Wielinga, R. de Hoog, H. Akkermans, and W. Van de Velde, "CommonKADS: A Comprehensive Methodology br KBS Development," *IEEE Expert Magazine*, vol. 9, no. 6, pp. 28 – 37, 1994. (cited on page 35)

[72] Object Modelling Group, "Universal Modelling Langauge 2.3," *Available online: "http://www.omg.org/spec/UML/2.3/"*. (cited on page 36)

[73] L. Zadeh, "Fuzzy Sets," *Information and Control*, vol. 8, pp. 338–353, Jan. 1965. (cited on page 39)

[74] T. Liu, J. H. Singonahallib, and N. R. Iyerb, "Detection of Roller Bearing Defects Using Expert System and Fuzzy Logic," *Mechanical Systems and Signal Processing*, vol. 10, pp. 595–614, Sept. 1996. (cited on page 39)

[75] F. Hayes-Roth and N. Jacobstein, "The State of Knowledge-Based Systems," *Communications of the ACM*, vol. 37, no. 3, pp. 26–39, 1994. (cited on page 40)

[76] Y. Shoham, "Agent-oriented programming," *Artificial Intelligence*, vol. 60, pp. 51–92, 1993. (cited on pages 42, 49)

[77] M. Wooldridge, C. Street, M. M, and N. R. Jennings, "Intelligent Agents : Theory and Practice," *Knowledge Engineering Review*, vol. 10, pp. 1–62, June 1995. (cited on page 42)

[78] N. R. Jennings and S. Bussmann, "Agent-Based Control Systems," *IEEE Control Systems Magazine*, pp. 61–73, June 2003. (cited on pages 43, 51)

[79] Foundation for Intelligent Physical Agents, "FIPA Standards Repository," *Available online at http://www.fipa.org/*. (cited on page 44)

[80] Foundation for Intelligent Physical Agents, "FIPA Agent Management Specification," *Available online at http://www.fipa.org/specs/fipa00023/*, 2004. (cited on pages 44, 45)

[81] Foundation for Intelligent Physical Agents, "FIPA ACL Message Structure Specification," *Available online at http://www.fipa.org/specs/fipa00061/*, 2002. (cited on page 45)

[82] Foundation for Intelligent Physical Agents, "FIPA Communicative Act Library Specification," *Available online at http://www.fipa.org/specs/fipa00037/*, 2002. (cited on page 46)

[83] Foundation for Intelligent Physical Agents, "FIPA Request Interaction Protocol Specification," *Available online at http://www.fipa.org/specs/fipa00026/*, 2002. (cited on page 46)

[84] J. Searle, *Speech acts: An essay in the philosophy of language*. Cambridge University Press, 1969. (cited on page 46)

[85] Foundation for Intelligent Physical Agents, "FIPA SL Content Language Specification," *Available online at http://www.fipa.org/specs/fipa00008/*, 2002. (cited on page 46)

[86] N. F. Noy and D. L. Mcguinness, "Ontology Development 101 : A Guide to Creating Your First Ontology," *Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, March 2001.* (cited on page 48)

[87] The Protégé Project, "Protégé," *available online at "http://protege.stanford.edu"*. (cited on page 48)

[88] A. Filipe, D. M. Batista, G. Bruno, T. Botelho, G. Kobayashi, P. Alves, and S. De, "Principles of Agent-Oriented Programming," in *Multi-Agent Systems - Modeling, Control, Programming, Simulations and Applications, Chapter 16*, pp. 317–342, 2011. (cited on page 49)

[89] A. S. Rao and M. P. Georgeff, "BDI Agents : From Theory to Practice," in *First International Conference on Multiagent Systems*, pp. 312–319, 1995. (cited on page 49)

[90] F. L. Bellifemine, G. Caire, and D. Greenwood, *Developing Multi-agent Systems with JADE*. John Wiley & Sons Ltd, 2007. (cited on page 49)

[91] S. D. J. McArthur, E. M. Davidson, V. M. Catterson, A. L. Dimeas, N. D. Hatziargyriou, F. Ponci, and T. Funabashi, "Multi-Agent Systems for Power Engineering Applications — Part I: Concepts, Approaches, and Technical Challenges," *IEEE Transactions on Power Systems*, vol. 22, pp. 1743–1752, Nov. 2007. (cited on page 50)

[92] S. D. J. McArthur, E. M. Davidson, V. M. Catterson, A. L. Dimeas, N. D. Hatziargyriou, F. Ponci, and T. Funabashi, "Multi-Agent Systems for Power Engineering Applications — Part II: Technologies, Standards, and Tools for Building Multi-agent Systems," *IEEE Transactions on Power Systems*, vol. 22, pp. 1753–1759, Nov. 2007. (cited on page 50)

[93] T. Wittig, N. R. Jennings, and E. H. Mamdani, "ARCHON : framework for intelligent co-operation," *Intelligent Systems Engineering*, vol. 3, no. 3, pp. 168–179, 1994. (cited on page 51)

[94] J. M. Corera, I. Laresgoiti, and N. R. Jennings, "Using Archon. 2. Electricity transportation management," *IEEE Expert*, pp. 71–79, 1996. (cited on page 51)

[95] S. Bussmann and K. Schild, "Self-Organizing Manufacturing Control : An Industrial Application of Agent Technology," in *Fourth International Conference on MultiAgent Systems*, pp. 87 – 94, 2000. (cited on page 51)

[96] J. Chang and K. Y. Lee, "Feedback control agents of the multiagent power plant control system," in *Proceedings of the 2003 International Conference on Machine Learning and Cybernetics*, pp. 1956–1960, 2003. (cited on page 52)

[97] K. Y. Lee, "Intelligent Techniques Applied to Power Plant Control," in *IEEE Power Engineering Society General Meeting.*, pp. 1–8, 2006. (cited on page 52)

[98] J. S. Heo and K. Y. Lee, "A Multiagent-System-Based Intelligent Reference Governor for Multiobjective Optimal Power Plant Operation," *IEEE Transactions on Energy Conversion*, vol. 23, pp. 1082–1092, Dec. 2008. (cited on page 52)

[99] E. Mangina, S. McArthur, J. McDonald, and A. Moyes, "A multi agent system for monitoring industrial gas turbine start-up sequences," *IEEE Transactions on Power Systems*, vol. 16, no. 3, pp. 396–401, 2001. (cited on page 52)

[100] S. McArthur, S. Strachan, and G. Jahn, "The Design of a Multi-Agent Transformer Condition Monitoring System," *IEEE Transactions on Power Systems*, vol. 19, pp. 1845–1852, Nov. 2004. (cited on page 52)

[101] V. M. Catterson and S. D. J. McArthur, "The Practical Implications of Bringing a Multi-Agent Transformer Condition Monitoring System On-Line," in *IEEE PES Power Systems Conference and Exposition*, pp. 12–16, 2004. (cited on page 52)

[102] S. E. Rudd, S. M. Strachan, and M. D. Judd, "An Incremental Knowledge Based Approach to the Analysis of Partial Discharge Data," in *15th International Symposium on High Voltage Engineering*, pp. 1–6, 2007. (cited on page 53)

[103] V. M. Catterson and S. D. J. McArthur, "Using Evidence Combination for Transformer Defect Diagnosis," *International Journal of Innovations in Energy Systems and Power*, vol. 1, no. 1, pp. 8–12, 2006. (cited on pages 53, 71, 150, 153)

[104] V. M. Catterson, S. E. Rudd, S. D. J. McArthur, and G. Moss, "On-line Transformer Condition Monitoring through Diagnostics and Anomaly Detection," in *2009 15th International Conference on Intelligent System Applications to Power Systems*, pp. 1–6, Nov. 2009. (cited on page 53)

[105] E. Davidson, S. McArthur, J. McDonald, T. Cumming, and I. Watt, "Applying Multi-Agent System Technology in Practice: Automated Management and Analysis of SCADA and Digital Fault Recorder Data," *IEEE Transactions on Power Systems*, vol. 21, pp. 559–567, May 2006. (cited on page 55)

[106] J. Hossack, J. Menal, S. D. J. McArthur, and J. R. McDonald, "A multiagent architecture for protection engineering diagnostic assistance," *IEEE Transactions on Power Systems*, vol. 18, pp. 639–647, May 2003. (cited on page 55)

[107] V. M. Catterson, E. M. Davidson, and S. D. J. McArthur, "Embedded Intelligent for Electrical Network Operation and Control," *IEEE Intelligent Systems*, vol. 26, no. 2, pp. 38–45, 2011. (cited on page 55)

[108] K. Y. Lee, J. S. Heo, J. a. Hoffman, S.-h. Kim, and W.-H. Jung, "Neural Network-Based Modeling for a Large-Scale Power Plant," in *IEEE Power Engineering Society General Meeting*, pp. 1–8, June 2007. (cited on pages 57, 152)

[109] Y. Luo, H. Liu, L. Jia, and W. Cai, "Modeling and Simulation of Ball Mill Coal-pulverizing System," in *IEEE Conference on Industrial Electronics and Applications*, pp. 1348–1353, 2011. (cited on pages 57, 60)

[110] G. Si, H. Cao, Y. Zhang, and L. Jia, "Application of information fusion based on RBF neural networks and fuzzy control to ball mill pulverizing system," in *7th World Congress on Intelligent Control and Automation*, pp. 4813–4818, 2008. (cited on pages 58, 60)

[111] A. E. Kukoski, "Ball Mill Pulverizer Design," in *Technology Applications Symposium*, 1992. (cited on page 58)

[112] Department of Trade and Industry, "Pulverised Fuel (PF) Flow Measurement and Control Methods for Utility Boilers," Tech. Rep. January, 2001. (cited on page 60)

[113] T.-Y. Chai and H. Yue, "Multivariable Intelligent Decoupling Control System," *Acta Automatica Sinica*, vol. 31, no. 1, pp. 123–131, 2005. (cited on page 60)

[114] L.-h. Feng, W.-h. Gui, and F. Yang, "Neural Network Modeling of Duplex Inlet and Outlet Ball Pulverizer System Based on Grey Relational Analysis," *2010 International Conference on Intelligent System Design and Engineering Application*, vol. 1, pp. 984–988, Oct. 2010. (cited on page 60)

[115] D. Julong, "Introduction to Grey System Theory," *The Journal of Grey System*, vol. 1, no. 1, pp. 1–24, 1989. (cited on page 60)

[116] R. G. Cheetham and S. A. Billings, "Mathematical modelling of power station plant - the role of simulation," tech. rep., 1988. Available online at `http://www.acslx.com/Support/TechPapers/Power/Power/63.pdf`; accessed 08-December-2012. (cited on pages 60, 67)

[117] J. Wang, W. Zhu, G. Si, and Y. Zhang, "Feature Extraction in Ball Mill Pulverizing System Based on Multi-sensor Data Fusion," in *2008 International Conference on Computer Science and Software Engineering*, pp. 573–576, 2008. (cited on page 60)

[118] G. Zhou, J. Si, and C. Taft, "Modeling and simulation of C-E deep bowl pulverizer," *IEEE Transactions on Energy Conversion*, vol. 15, no. 3, pp. 312–322, 2000. (cited on page 60)

[119] C. Booth and J. R. Mcdonald, "The use of artificial neural networks for condition monitoring of electrical power transformers," *Neurocomputing*, vol. 23, pp. 97–109, 1998. (cited on page 71)

[120] J. Benesty, S. Member, J. Chen, and Y. A. Huang, "On the Importance of the Pearson Correlation Coefficient in Noise Reduction," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, pp. 757–765, May 2008. (cited on page 74)

[121] M. P. Boyce, *Gas Turbine Engineering Handbook*. Gulf Professional Publishing, third ed., 2006. (cited on pages 79, 80, 81, 82, 83)

[122] A. Giampaolo, *Gas Turbine Handbook Principles and Practices*. Fairmont Press, 3rd ed., 2006. (cited on page 80)

[123] Emerson Process Management, "Gas Turbine Engine Performance - Evaluating the Impact of Cyclic Maintenance Strategies," Tech. Rep. January, 2005. available online at http://www2.emersonprocess.com/siteadmincenter/PM%20Asset%20Optimization%20Documents/ProductWhitePapers/amspm_wp_GasTurbinePerf.pdf; accessed 20 December 2012. (cited on page 80)

[124] R. Kurz and K. Brun, "Gas Turbine Tutorial - Maintenance and Operating Practices Effects on Degradation and Life," in *36th Turbomachinery Symposium*, pp. 173–185, 2007. (cited on page 82)

[125] M. Maalouf, "Gas Turbine Vibration Monitoring - An Overview," *Orbit*, vol. 25, no. 1, pp. 48–62, 2005. (cited on page 83)

[126] M. R. B. Tavakoli, B. Vahidi, and W. Gawlik, "An Educational Guide to Extract the Parameters of Heavy Duty Gas Turbines Model in Dynamic Studies Based on Operational Data," *IEEE Transactions on Power Delivery*, vol. 24, no. 3, pp. 1366–1374, 2009. (cited on page 83)

[127] C. E. Shannon, "A mathematical theory of communication," *Bell Systems Technical Journal*, vol. 27, pp. 379–423, 623–656., 1948. (cited on pages 83, 89)

[128] Beran Instruments, "PlantProtech," *available online at www.beraninstruments.co.uk*. (cited on pages 83, 88, 89)

[129] S. Gupta and A. Ray, "Symbolic Dynamic Filtering," in *Pattern Recognition: Theory and Application*, ch. 2, pp. 17–71, 2007. (cited on page 83)

[130] S. W. Wegerich, "Similarity based modeling of time synchronous averaged vibration signals for machinery health monitoring," *2004 IEEE Aerospace Conference Proceedings*, pp. 3654–3662, 2004. (cited on pages 84, 90)

[131] G. Peng, "WEC Condition Monitoring Based on SCADA Data Analysis," in *Proceedings of the 30th Chinese Control Conference*, pp. 5099–5103, 2011. (cited on page 84)

[132] D. A. Clifton, L. Tarassenko, N. Mcgrogan, D. King, S. King, and P. Anuzis in *IEEE Aerospace Conference*, pp. 1–11, Mar. 2008. (cited on page 84)

[133] W. I. Rowen, "Simplified Mathematical Representations of Heavy-Duty Gas Turbines," *Journal Of Engineering For Power*, vol. 105, no. 83, pp. 865–869, 1984. (cited on page 84)

[134] S. K. Yee, J. V. Milanovic, and F. M. Hughes, "Overview and Comparative Analysis of Gas Turbine Models for System Stability Studies," *IEEE Transactions on Power Systems*, vol. 23, no. 1, pp. 108–118, 2008. (cited on page 84)

[135] Z. Abu-el zeet and V. C. Patel, "Power Plant Condition Monitoring Using Novelty Detection," in *International Conference on Systems Engineering*, pp. 9–14, 2006. (cited on pages 84, 85)

[136] P. Smyth, "Hidden Markov models and neural networks for fault detection in dynamic systems," *Neural Networks for Signal Processing III - Proceedings of the 1993 IEEE-SP Workshop*, pp. 582–592, 1993. (cited on page 84)

[137] J. Lee, S. Kim, Y. Hwang, and C. Song, "Diagnosis of mechanical fault signals using continuous hidden Markov model," *Journal of Sound and Vibration*, vol. 276, pp. 1065–1080, Sept. 2004. (cited on page 84)

[138] V. Purushotham, S. Narayanan, and S. Prasad, "Multi-fault diagnosis of rolling bearing elements using wavelet analysis and hidden Markov model based fault recognition," *NDT & E International*, vol. 38, pp. 654–664, Dec. 2005. (cited on page 85)

[139] J. Huang and P. Zhang, "Fault Diagnosis for Diesel Engines Based on Discrete Hidden Markov Model," *2009 Second International Conference on Intelligent Computation Technology and Automation*, pp. 513–516, 2009. (cited on page 85)

[140] R. Milne, "TIGER : Knowledge Based Gas Turbine Condition Monitoring," in *IEE Colloquium on Artificial Intelligence-Based Applications for the Power Industry*, pp. 5/1–5/4, 1999. (cited on pages 85, 86)

[141] R. Milne, "TIGER: numeric and qualitative model based diagnosis," in *IEE Colloquium on 'Qualitative and Quantitative Modelling Methods for Fault Diagnosis'*, pp. 3/1 – 3/7, 1995. (cited on page 87)

[142] K. Bousson and L. Trave-massuyes, "Fuzzy causal simulation in process engineering," in *Proc. 13th Int. Joint Conf. on Artificial Intelligence*, pp. 35–44, 1993. (cited on page 87)

[143] M. Ghallab and H. Philippe, "A Compiler for Real-Time Knowledge-Base Systems," in *International Workshop on Artificial Intelligence for Industrial Applications*, pp. 387–393, 1988. (cited on page 87)

[144] C. Dousson, P. Gaborit, and M. Ghallab, "Situation Recognition : Representation and Algorithms," in *Proc. 13th UCAI, Chambery, France*, pp. 166–172, 1993. (cited on page 87)

[145] P. Laborie and M. Ghallab, "IxTeT : an Integrated Approach for Plan Generation and Scheduling," in *Proceedings of INRIA/IEEE Symposium on Emerging Technologies and Factory Automation.*, pp. 485 – 495, 1995. (cited on page 87)

[146] "SmartSignal Predictive Diagnostics," *Available online at* `http://www.ge-mcs.com/en/bently-nevada-software/process-modeling-optimization-and-control/smart-signal.html`; *accessed 17 December 2012.* (cited on page 90)

[147] M. Todd, S. D. J. McArthur, J. R. Mcdonald, and S. J. Shaw, "Generator Diagnostic Knowledge," *IEEE Transactions on Systems. Man, and Cybernetics - Part C: Applications and Reviews*, vol. 37, no. 5, pp. 979–992, 2007. (cited on page 99)

[148] K. Murphy, "Hidden Markov Model (HMM) Toolbox for Matlab," *Available online at http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html*, 1998. (cited on pages 102, 108, 174)

[149] IEEE Computer Society, "IEEE Std 754-2008 (Revision of IEEE Std 754-1985), IEEE Standard for Floating-Point Arithmetic," 2008. (cited on page 117)

[150] V. M. Catterson and S. D. J. McArthur, "Off-Shore Wind Condition Monitoring: The Challenges and Research Potential," tech. rep., 2010. (cited on pages 134, 135)

[151] V. M. Catterson, E. M. Davidson, and S. D. J. McArthur, "Issues in Integrating Existing Multi-Agent Systems for Power Engineering Applications," in *Proceedings of the 13th International Conference on, Intelligent Systems Application to Power Systems*, pp. 396–401, 2005. (cited on page 134)

[152] V. M. Catterson, S. D. J. McArthur, S. Member, and G. Moss, "Online Conditional Anomaly Detection in Multivariate Data for Transformer Monitoring," *IEEE Transactions on Power Delivery*, vol. 41, pp. 2556–2564, Oct. 2010. (cited on page 134)

[153] Distributed Management Task Force, "Common Information Model Infrastructure Specification Version 2.3 Final," 2005. (cited on page 134)

[154] G. Shafer, "Combination of Evidence," *International Journal of Intelligent Systems*, vol. 1, pp. 155–179, 1986. (cited on page 150)

[155] G. C. Stone, "Advancements during the past quarter century in on-line monitoring of motor and generator winding insulation," *IEEE Transactions on Dielectrics and Electrical Insulation*, vol. 9, pp. 746–751, Oct. 2002. (cited on page 151)

[156] S. Simani, "Identification and Fault Diagnosis of a Simulated Model of an Industrial Gas Turbine," *IEEE Transactions on Industrial Informatics*, vol. 1, pp. 202–216, Aug. 2005. (cited on page 153)

[157] A. P. Dempster, "Upper and Lower Probabilities Induced by a Multivalued Mapping," *Annals of Mathematical Statistics*, vol. 38, no. 2, pp. 325–339, 1967. (cited on page 153)

[158] J.-b. Yang and M. G. Singh, "An Evidential Reasoning Approach for Multiple- Attribute Decision Making with Uncertainty," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 24, no. I, pp. 1 – 18, 1994. (cited on page 153)

[159] J.-b. Yang and D.-l. Xu, "On the Evidential Reasoning Algorithm for Multiple Attribute Decision Analysis Under Uncertainty," *IEEE Transactions on Systems. Man, and Cybernetics - Part A: Systems and Humans*, vol. 32, no. 3, pp. 289–304, 2002. (cited on page 153)

[160] R. Liao, H. Zheng, S. Grzybowski, L. Fellow, L. Yang, Y. Zhang, and Y. Liao, "An Integrated Decision-Making Model for Condition Assessment of Power Transformers Using Fuzzy Approach and Evidential Reasoning," *Power Delivery, IEEE Transactions on*, vol. 26, no. 2, pp. 1111–1118, 2011. (cited on page 153)

[161] J. Bilmes, "What HMMs Can Do," Tech. Rep. 206, University of Washington, 2002. (cited on page 174)