Master of Science Thesis in Electrical Engineering
Department of Electrical Engineering, Linköping University, 2020

Data-driven engine fault classification and severity estimation using residuals and data

Andreas Lundgren

LINKÖPING
UNIVERSITY

Master of Science Thesis in Electrical Engineering

Data-driven engine fault classification and severity estimation using residuals and data:

Andreas Lundgren

LiTH-ISY-EX--20/5289--SE

Supervisor: **Sergii Voronov**
ISY, Linköpings universitet

Examiner: **Daniel Jung**
ISY, Linköpings universitet

Division of Vehicular Systems
Department of Electrical Engineering
Linköping University
SE-581 83 Linköping, Sweden

Abstract

Recent technological advances in the automotive industry have made vehicular systems increasingly complex in terms of both hardware and software. As the complexity of the systems increase, so does the complexity of efficient monitoring of these system. With increasing computational power the field of diagnostics is becoming evermore focused on software solutions for detecting and classifying anomalies in the supervised systems. Model-based methods utilize knowledge about the physical system to device nominal models of the system to detect deviations, while data-driven methods uses historical data to come to conclusions about the present state of the system in question. This study proposes a combined model-based and data-driven diagnostic framework for fault classification, severity estimation and novelty detection.

An algorithm is presented which uses a system model to generate a candidate set of residuals for the system. A subset of the residuals are then selected for each fault using L1-regularized logistic regression. The time series training data from the selected residuals is labelled with fault and severity. It is then compressed using a Gaussian parametric representation, and data from different fault modes are modelled using 1-class support vector machines. The classification of data is performed by utilizing the support vector machine description of the data in the residual space, and the fault severity is estimated as a convex optimization problem of minimizing the Kullback-Leibler divergence (KLD) between the new data and training data of different fault modes and severities.

The algorithm is tested with data collected from a commercial Volvo car engine in an engine test cell and the results are presented in this report. Initial tests indicate the potential of the KLD for fault severity estimation and that novelty detection performance is closely tied to the residual selection process.

Sammanfattning

Tekniska innovationer inom fordonsindustrin har på senare tid lett till att moderna fordonssystem är mer avancerade än någonsin tidigare, både vad gäller hårdvara och mjukvara. När komplexiteten hos systemen ökar, ökar även komplexiteten i problemet att designa effektiva diagnossystem. Med ökande fordonsbunden beräkningskapacitet har fokus inom fältet alltmer kommit att skiftas mot mjukvarulösningar för att detektera och klassificera anomalier i det övervakade systemen. Modellbaserade metoder utnyttjar insikter om systemets fysikaliska egenskaper för att skapa nominella modeller av systemet och använder dessa för att upptäcka avvikelser från normala beteenden. Datadrivna metoder använder historiska data från det övervakade systemet för att dra slutsatser om dess nuvarande funktion. I denna studie föreslås ett diagnosramverk som kombinerar datadrivna och modellbaserade metoder för fel-klassificering, felstorleks-estimering och *outlier*-detektion.

En algoritm, som använder en systemmodell för att generera en mängd kandidatresidualer, presenteras. En delmängd av dessa kandidater är sedan valda för varje feltyp genom L1-regulariserad logistisk regression. Tidsserie-data från de valda residualerna kategoriseras med feltyp och felstorlek. Denna är sedan komprimerad med hjälp av en Gaussisk parameterrepresentation, och varje feltyp modelleras med en *1-class support vector machine* (SVM). Dataklassificeringen utförs genom att använda SVM-representationen i de residual-rum som konstruerats för varje feltyp. Estimeringen av felstorlek är given som lösningen till ett konvext optimeringsproblem som minimerar Kullback-Leibler-divergensen mellan den nya residualdatan och träningsdatan från olika feltyper och felstorlekar.

Algoritmen är testad på data som har samlats in från en kommersiell förbränningsmotor från Volvo i en motortestcell och dessa resultat finns presenterade i denna rapport. Inledande tester påvisar potentialen hos att använda Kullback-Leibler-divergensen för felstorleksestimering och att prestandan hos *outlier*-detektionen är tätt knuten till resultaten från residualvalsprocessen.

Acknowledgments

This work is part of the research conducted at the division of Vehicular Systems in the department of Electrical Engineering at Linköping University.

I would like to thank Sergii Voronov who acted as supervisor and through his input helped shape the project. I would also like to thank my examiner Daniel Jung who, through this continuous support and feedback, played an invaluable part in the completion of this work. Lastly a special thanks should be extended to Tobias Lindell, without whom the data collection would not have been possible.

Linköping, March 2020
Andreas Lundgren

Contents

Notation **xi**

1 Introduction **1**
1.1 Motivation . . . 2
1.1.1 Fault severity estimation . . . 2
1.1.2 Data dimensionality . . . 2
1.1.3 Data collection and representation . . . 3
1.2 Purpose . . . 3
1.3 Research questions . . . 3
1.4 Delimitations . . . 4
1.5 Methodology . . . 4

2 Theoretical framework **5**
2.1 Kullback-Leibler divergence . . . 5
2.2 Model-based diagnostics . . . 6
2.3 Data-driven diagnostics . . . 8
2.3.1 Support vector machines . . . 9
2.3.2 Logistic regression . . . 14

3 Proposed algorithm **17**
3.1 Residual selection . . . 17
3.2 Data processing . . . 19
3.3 Data classification . . . 21
3.3.1 Partial- and final diagnosis . . . 22
3.4 Fault severity estimation . . . 23
3.5 Full algorithm . . . 26

4 Results **29**
4.1 Data collection . . . 29
4.2 Classifier training . . . 32
4.3 Data classification . . . 36

5 Discussion **41**

5.1 Result 41
5.2 Methodology 42

6 Conclusions 45
6.1 Future work 46

Bibliography 47

Notation

MATHEMATICAL NOTATION

Notation	Meaning
$\mathcal{K}(p\|\|q)$	Kullback-Leibler divergence of density functions p & q.
$\mathcal{L}(\,\cdot\,)$	The Lagrange function
f_i	Fault mode of type i
T_{f_i}	Classifier designed to detect f_i

ABBREVIATIONS

Abbreviation	Meaning
FDD	Fault detection and diagnosis
KLD	Kullback-Leibler divergence
NF	Fault free mode
SVM	Support vector machine
WLTC	Worldwide harmonized Light-duty vehicles Test Cycles

1 Introduction

Recent technological advances in the automotive industry have made modern vehicular systems more complex than ever before. It is of the utmost importance that these systems are functioning as intended and to any unnecessary breakdowns or associated risks. One important part avoiding any such breakdowns is the ability to detect and isolate any faults in the systems so that these can be addressed before they cause additional damage. A part of this is the design of accurate diagnostic systems. Traditionally, these systems have, to a large degree, been based on adding hardware redundancy to the vehicles in order to make the systems more robust against component failure. For example, if two sensors are measuring the same parameter and they show two different values, one might assume that there is something wrong with either the sensors or with the system which they are monitoring. This is an expensive solution as it might mean that, in order to be able to perform system diagnostics, the system might need significantly more sensors than are needed for general operations. As the onboard computational power has increased, an effort has been put into developing alternative solutions to this problem. Software solutions have been increasingly used and thus reducing the need of redundant hardware. This study can be seen as yet another contribution to this field.

The purpose of this document is to describe the design, implementation and validation process of a hybrid data-driven and model-based diagnosis system for an internal combustion engine. Chapter 1 gives an introduction to the problem and purpose of the project. The central theoretical concepts that are used in the system are explained in Chapter 2. In Chapter 3, the proposed solution is described in closer detail and some experimental results are presented in Chapter 4. The thesis is wrapped up with a discussion about the findings in Chapter 5 followed by some concluding remarks in Chapter 6.

1.1 Motivation

This section gives an overview of some of the current problems that this study addresses as well as some of the research related to this topic.

1.1.1 Fault severity estimation

A large part of the existing work done on fault detection and diagnosis (FDD) is concerned with fault detection and classification. An overview of some methods that are used can be found in [37–39]. There is however one area that is largely unexplored, and that is data-driven fault size estimation. Rather than only classifying any occurring faults it is also of interest to have an idea of the magnitude or severity of these faults, especially from a maintenance point of view. When deciding whether or not a system needs maintenance, information about if components are slightly degraded or close to failure alleviates the decision process, which in turn reduces the cost of unnecessary maintenance and unforeseen breakdowns. Some work has been done on the topic, mainly on bearings [13, 31]. Both of these methods are, however unsuited to apply in the combustion engine case. In [31], faults are assumed to appear as pulses in the time-domain data which is inherently tied to the bearing case, and [13] uses Paris' formula [28] to interpolate between distributions from known fault sizes. This assumption can not be reasonably extended to hold for general mechanical faults or sensor faults.

One approach that has recently received attention, is using the Kullback-Leibler divergence (KLD) as a diagnostic tool [16–18, 41, 42]. These works have all been focused on the FDD problem for incipient faults and size estimation using analytic expressions based on assumptions about fault characteristics. Another solution is proposed by [44], where the idea is to find a parameter space such that each kind of fault realizations appear separate lines and the distance to these lines is used to classify new data.

1.1.2 Data dimensionality

Working with high-dimensional data, there are some phenomena which might not be as prevalent in lower dimensions. This is what is often referred to as the curse of dimensionality [40]. As the number of dimensions grow, so does the number of measurements required to maintain sufficient observation density, which is defined as "observations per volumetric unit in the observation space". What might happen otherwise is that slight differences will cause the data to appear sparse so that any accurate classification becomes problematic. For that reason, it is of interest to reduce the dimensionality of the data in a way that maintains sufficient information for classification.

The problem of the dimensionality of data representation is also relevant when it comes to practical aspects of the design of large scale diagnostic systems, and the split between onboard and offboard diagnostics. In the case of onboard systems, reference data needs to be stored in the system memory and representation is important to reduce the amount of memory space needed for accurate diagnostics. In the case of offboard diagnosis, the vehicle can either store log data which is later analyzed in, for example, a workshop in which case memory is an issue. Offboard diagnostics could also be performed by transmitting data to a remote location where the actual diagnostics is handled, in which case the bandwidth of the channel has to be taken into account.

1.1.3 Data collection and representation

One of the most important requirements when it comes to data-driven FDD is the availability of representative training data. Ideally, there would be data available for every possible fault mode and severity, which is not possible since data collection from structured tests is a time consuming process, and the number of known possible faults increases rapidly with the complexity of the system. Besides the tested faults, other issues that the designer did not consider might also arise. To continuously improve the model, new data has to be added and explained. Such data can naturally be obtained when the vehicle in question is in for service. The sensor data is collected from the onboard computer and is labelled by the mechanic after the diagnostic. The problem with this is the amount of data that needs to be transferred from the car can be substantial. From a performance perspective, the vehicle should store as little data as possible to save memory space. On the other hand, diagnosis gets easier the more data that is available. For this reason it is of interest to see if it is possible to reduce the amount of information needed for the diagnosis. If residual data, representative of a faulty state, can be expressed as a known type of probability distribution, the entire set could be described by a few parameters rather than having to send the entire set of sensor readings. This requires that the loss of information does not reduce performance below an acceptable limit.

1.2 Purpose

The purpose of this study is to develop a diagnostic framework for fault severity estimation in a combustion engine. The goal is to device a diagnostic framework that is able to utilize the data from model-based classifiers and use machine learning based techniques to improve their performance. Another part of the study is to see how the residual data should be represented so as to minimize the amount of stored data while at the same time maintaining diagnostic performance.

1.3 Research questions

To address the problems described in Section 1.1, this study focuses on three areas of interest: severity estimation, novelty detection and data representation. All these topics are considered in an environment where the availability of training data and storage is limited, and thus has to be taken into account in the design of any diagnostic framework. To capture these problems, the following research questions were formulated, where the goal of each question is to capture a specific one of the listed areas of interest.

1. Can limited historical data of known fault type and severity be utilized to recognize, and estimate the severity of, new faults to prevent breakdowns and/or unnecessary maintenance?

2. How can a classifier be designed to distinguish between data from a known fault mode, which is represented in training data, and data from a previously unknown fault?
3. How can onboard data be represented to reduce the bandwidth, or storage of operation data, required for offboard diagnostics?

1.4 Delimitations

To focus the study on the stated research questions, the following delimitations were set:

- Residuals are generated using an existing model and further engine modelling design is not a part of this study.
- Residual construction and selection are problems that have received considerable attention in their own right, for example [24] [23]. These considerations are left out of the study and residuals, along with corresponding decision matrices are already constructed.
- The available training data is assumed to be labelled so that the fault type and severity is specified for each data set:

1.5 Methodology

This work is conducted as part literature study and part experimental work. Firstly, an overview of the field is provided along with some of the available work related to the posed problem. Secondly, a new method is proposed together with a theoretical discussion about the rationale behind it. The suggested method is tested on empirical data, collected for the purpose of this work. The data is collected in a laboratory setting on a modified commercial system. The work is concluded with a discussion about the findings and their validity.

2

Theoretical framework

In this section, some of the studies central theoretical concepts are introduced. The mathematical definitions are presented here along with central definitions, and important properties are discussed. The specific application of these methods, in relation to this study, are generally not presented. This is done in Chapter 3 were the implementation is discussed in detail. Instead, this chapter should be seen as a general introduction to these concepts and as a background to the contribution of this study.

2.1 Kullback-Leibler divergence

In statistical applications, observation data is often presented as realizations of a random variable. The observations conform to a pattern with some added noise and the behaviour could be described using a probability distribution. Different type of data can be described by different types of distributions. Given more than one of these data sets, it is often relevant to tell how different these distributions are. As mentioned in Section 1.1, the Kullback-Leibler divergence (KLD) is one way of measuring the similarity, or rather dissimilarity, between two distributions. It is based on the Kullback-Leibler information [26] of two random variables.

Definition 2.1 (Kullback-Leibler divergence). Let P and Q be continuous random variables with probability density functions $p(x)$ and $q(x)$ respectively. The Kullback-Leibler divergence between $p(x)$ and $q(x)$ is defined as

$$\mathcal{K}(p\|q) = \int p(x) \log\left(\frac{p(x)}{q(x)}\right) dx. \tag{2.1}$$

Although the KLD measures the similarity between distributions, it is not a metric since it does not necessarily satisfy the triangle inequality and is generally asymmetric i.e. $\mathcal{K}(p\|q) \neq \mathcal{K}(q\|p)$. It is however a non-negative measure as $\mathcal{K}(p\|q) \geq 0$ with equality only in the case $\mathcal{K}(p\|p) = 0$. Broadly speaking it can be said that $\mathcal{K}(p\|q)$ should be close to 0 if p and q are similar and increase the more dissimilar they are.

There are instances when the KLD can be expressed analytically. One such case is the KLD between two Gaussian distributions. If $\mathcal{N}_0, \mathcal{N}_1$ are two k-dimensional normal distributions with mean μ_0, μ_1 and covariance matrices Σ_0, Σ_1 the KLD becomes [6]

$$\mathcal{K}(\mathcal{N}_0\|\mathcal{N}) = \frac{1}{2}\left(\mathrm{Tr}\left(\Sigma_1^{-1}\Sigma_0\right) + (\mu_1 - \mu_0)^\intercal \Sigma_1^{-1}(\mu_1 - \mu_0) - k + \ln\left(\frac{\det \Sigma_1}{\det \Sigma_0}\right)\right). \tag{2.2}$$

The KLD between Gaussian mixture models on the other hand, has no such analytical expression, according to [20]. There are several different methods that approximate the KLD numerically. For a comparison of different approximation methods see [20]. In this study Monte Carlo sampling has been used as an approximation method. The KLD function is estimated by generating a large number n of samples $\{x_i\}_{i=1}^n$ from $p(x)$ to replace the integration in (2.1) with the following sum

$$\mathcal{K}_{MC}(p\|q) = \frac{1}{n}\sum_{1=1}^{n} \ln\left(\frac{p(x_i)}{q(x_i)}\right). \tag{2.3}$$

By the law of large numbers $\lim_{n\to\infty} \mathcal{K}_{MC}(p\|q) = \mathcal{K}(p\|q)$. A closer examination of the actual upper and lower bounds of this approximation is found in [7].

2.2 Model-based diagnostics

One way of conducting diagnosis is to use a model-based approach. A model of the physical system is produced and this model is then used in conjunction with data $z[t]$ from the real system to conclude if the current state of the system is consistent with the nominal model. One way of doing this is by defining so called consistency relations. These are relations that should hold in a fault free environment.

Example 2.2: Consistency relations

To exemplify this, consider the example of monitoring the pressure in an empty tank. The inflow of air is denoted f_{in}, the outflow is denoted f_{out} and the pressure is denoted p. Due to the dynamics of the system, the change in pressure $\dot{p}$ should be proportional to the difference in inflow and outflow giving the follow-

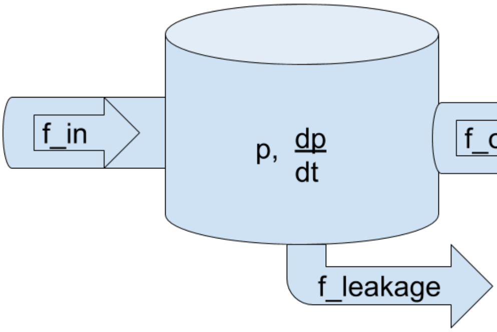


Figure 2.1: *Example system for leakage detection in an empty tank.*

ing relation, $\dot{p} = \alpha(f_{in} - f_{out})$. Assuming that the variables p, f_{in} and f_{out} are all measurable signals, the following consistency relation can be constructed

$$\dot{p} - \alpha(f_{in} - f_{out}) = 0. \tag{2.4}$$

If we now assume that the system is not fault free and that there is a hole in the tank. Let the airflow through this hole be denoted $f_{leakage}$. A schematic view of this system is shown in Figure 2.1. The true system would be described by the relation $\dot{p} = \alpha(f_{in} - f_{out} - f_{leakage})$ and Eq. (2.4) does no longer hold. We can use this fact to construct a function $r_0 = \dot{p} - \alpha(f_{in} - f_{out})$ and say that if $r \neq 0$ there is a leakage, or more generally

$$r = \begin{cases} 0 & \text{NF} \\ \neq 0 & \text{otherwise} \end{cases} \tag{2.5}$$

Definition 2.3 (Residual generators). A function $r(z)$ is said to be a residual generator if $r(z) = 0$ for a fault free system. The output of $r(z)$ is called a residual.

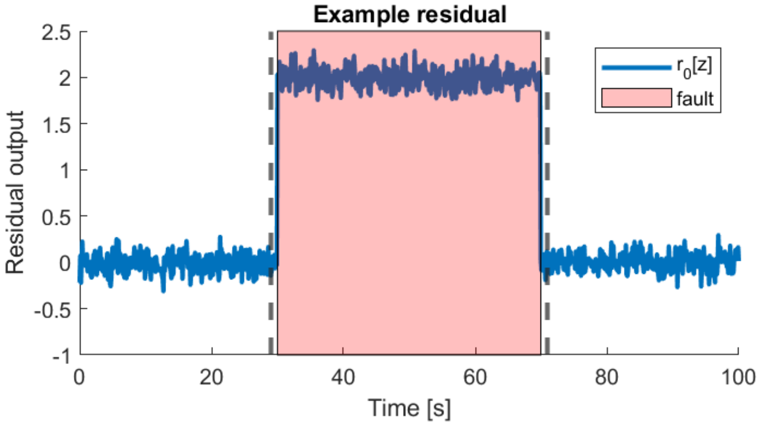


***Figure 2.2:** Illustration of an example residual.*

In the case of a system with multiple potential faults, one important property of the residual generator is for which of the fault it reacts i.e. is non-zero.

Definition 2.4 (Sensitivity). A residual generator is said to be sensitive to fault f_i if $r(z) \neq 0$ implies that f_i is present in the system.

An example residual is shown in Figure 2.2. The residual r_0 is subject to some noise but is close to zero in the nominal system. When a fault, which the residual is sensitive to, occurs in the system the residual becomes noticeably non-zero only to return to zero when the fault is removed.

For this example system, creating a residual generator was quite straight forward, but as the complexity of the system increases so does the complexity of the problem of residual generator design. These problems have gotten considerable attention but will not be further examined here. Some details can be found in previously mentioned [23, 24].

2.3 Data-driven diagnostics

An alternative to model-based diagnostics is using a data-driven approach. Rather than using a physics-based model of the system, the idea is to use large amounts of historical data to characterize the monitored system; a method referred to as machine learning. This is a field that currently receives a massive interest in a variety of applications. An overview of some of the machine learning research related to FDD specifically can be found in [3, 14, 21, 22, 30, 38].

The methods that will be discussed in this study are support vector machines for classification and logistic regression for feature extraction.

2.3.1 Support vector machines

Support vector machines is one method that is used for supervised learning [35]. To get an understanding of support vector machines, first consider the traditional two-class support vector machine. A set of labeled training data from two different classes are assumed to be available and the objective is to create a classifier such that new observations, from either of these classes, are associated with the correct class. Denote the training data as $\omega = \{(x_1, y_1) \ldots (x_n, y_n)\}$ where $x \in \mathbb{R}^k$ is a k-dimensional data vector and $y \in \{-1, 1\}$ denotes which class each observation belongs to. The following notation is used to distinguish between data from the different classes: $x_i \in C_j$ means that x_i belongs to the class associated with $y_i = j$. Then create a classifier $h(x)$ such that

$$h(x) = \begin{cases} 1, & \text{if } x \in C_1 \\ -1, & \text{if } x \in C_{-1} \end{cases} \tag{2.6}$$

This can be accomplished in different ways depending on how the data is distributed, but consider first the case where there exist a hyperplane, which separates the data, so that all points belonging to C_1 are on one side of the hyperplane and the points belonging to C_{-1} are on the other side of this hyperplane. Note that any of these hyperplanes can be expressed as $\omega^\mathsf{T} x - b = 0$, where ω is a normal vector to the hyperplane and b is a constant. This gives a potentially infinite set of hyperplane candidates so some added criteria are needed to uniquely define the optimal hyperplane. This can be done by creating two parallel hyperplanes and maximize the distance between them. The region between the hyperplanes is called the margin and the dividing hyperplane is in the middle of this margin. This is illustrated for the two-dimensional case ($x \in \mathbb{R}^2$) in Figure 2.3.

To construct this margin, let the two planes be defined as

$$\omega^\mathsf{T} x - b = 1 \tag{2.7}$$

$$\omega^\mathsf{T} x - b = -1 \tag{2.8}$$

The distance between the hyperplanes is $2/\|\omega\|$. Thus, the goal is to minimize $\|\omega\|$ while still maintaining all elements on the correct side of the hyperplane and to make sure that no elements are placed within the margin. These conditions can be summarized as the following optimization problem:

$$\begin{aligned} \min_{\omega, b} \quad & \|\omega\| \\ \text{s.t.} \quad & y_i(\omega^\mathsf{T} x_i - b) \geq 1 \quad \text{for } i = 1 \ldots n \end{aligned} \tag{2.9}$$

If ω^* and b^* are a solution to this problem, the hyperplane is chosen as $\omega^{*\mathsf{T}} x - b^* = 0$, and the resulting classifier can be written as

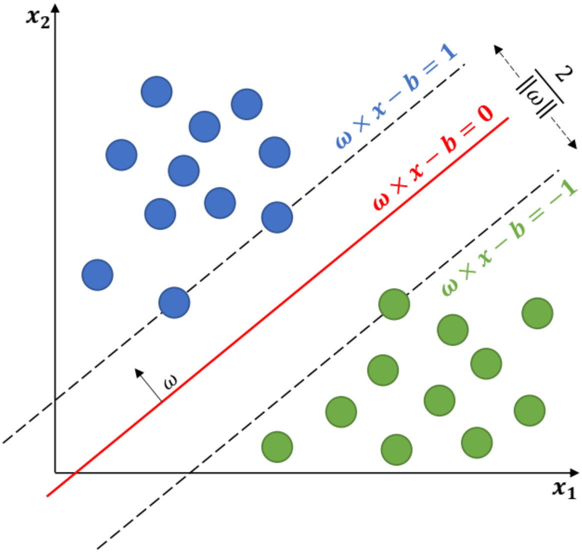

x_2
x_1
$\omega \times x - b = 1$
$\omega \times x - b = 0$
$\omega \times x - b = -1$
$\frac{2}{\|\omega\|}$
ω

Figure 2.3: *Illustration of the SVM hyperplane separating two linearly separable classes in 2-D.*

$$h(x) = \text{sgn}(\omega^{*\mathsf{T}} x - b^*) \tag{2.10}$$

As stated above, these conditions do not allow for any points from the training data to be placed inside the margin. To reduce the risk of overfitting these conditions can be loosened to allow for some of the training points to be placed within the margin. Rather than a "hard" margin a "soft" margin is created with the addition of slack variables ξ_i to allow some points to lie within the margin and a penalty variable $C > 0$ is added to adjust the trade-off between minimizing $\|\omega\|$ and the number of points within the margin. The updated problem becomes

$$\begin{aligned} \min_{\omega,\, b} \quad & \frac{1}{2}\|\omega\| + C\sum_i \xi_i \\ \text{s.t.} \quad & y_i(\omega^\mathsf{T} x_i + b) \geq 1 - \xi_i \quad i = 1 \ldots n, \\ & \xi_i \geq 0 \qquad\qquad\qquad\; i = 1 \ldots n \end{aligned} \tag{2.11}$$

This is a convex optimization problem and can be solved using quadratic programming. To study this problem the Lagrange function $\mathcal{L}$ is introduced as follows

$$\mathcal{L}(\omega, b, \xi, \alpha, \beta) = \frac{1}{2}\|\omega\|^2 + C\sum_i \xi_i - \sum_i \alpha_i\Big(y_i((\omega \cdot x_i) + b) - 1 + \xi_i\Big) - \sum_i \beta_i \xi_i \quad (2.12)$$

where $\alpha = (\alpha_1 \dots \alpha_n)^\mathsf{T}$ and $\beta = (\beta_1 \dots \beta_n)$ are called the Lagrange multipliers. As shown in [5], this means that the dual problem of Eq. (2.11) can be expressed as

$$\begin{aligned}
\max_{\alpha, \beta} \quad & -\frac{1}{2}\sum_i \sum_j y_i y_j \alpha_i \alpha_j (x_i \cdot x_j) + \sum_j \alpha_j \\
\text{s.t.} \quad & \sum_i y_i \alpha_i = 0, \\
& C - \alpha_i - \beta_i = 0 \quad i = 1 \dots n, \\
& \alpha_i \geq 0 \quad i = 1 \dots n, \\
& \beta_i \geq 0 \quad i = 1 \dots n
\end{aligned} \quad (2.13)$$

By eliminating using the second condition and letting $\beta_i = C - \alpha_i$ the problem can be rewritten as

$$\begin{aligned}
\max_{\alpha} \quad & \sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j y_i y_j \alpha_i \alpha_j (x_i \cdot x_j) \\
\text{s.t.} \quad & \sum_i y_i \alpha_i = 0, \\
& 0 \leq \alpha_i \leq C \quad i = 1 \dots n.
\end{aligned} \quad (2.14)$$

The reason that the solution to this problem is relevant to solving the original problem Eq (2.11) is that it has the following property, as seen in [5]. Let $\alpha^* = (\alpha_1^* \dots \alpha_n^*)$ denote a solution to the dual problem Eq. (2.14). The optimal solution (ω^*, b^*) to the primal problem Eq. (2.11) is then given by

$$\omega^* = \sum_i \alpha_i^* y_i x_i \quad (2.15)$$

$$b^* = y_j - \alpha_i^* y_i (x_i \cdot x_j). \quad (2.16)$$

Note that this expression only depends on points x_i corresponding to multipliers $\alpha_i \neq 0$. These points are said to "support" the solution, hence the term support vector machines.

The methods mentioned above are all examples of binary (2-class) classification. All data is classified as belonging to either class 1 or class 2. A related, but slightly different form of classification, is 1-class classification. In this case a classifier is constructed to model data from a single fault with the purpose of deciding

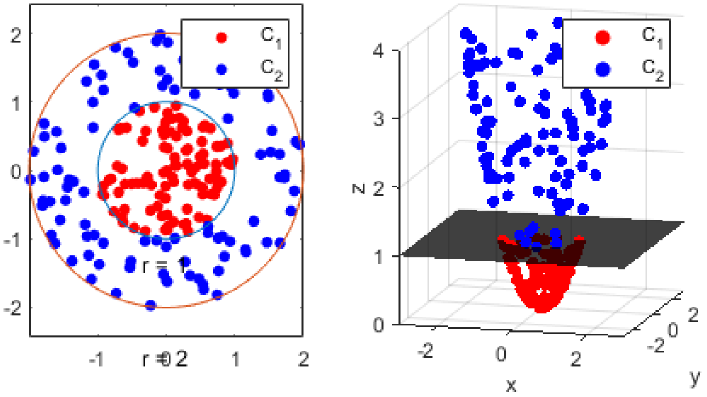


__Figure 2.4:__ Illustration of 2-D data being transformed into a 3-D space in which the two classes C_1 and C_2 are linearly separable. The transformation is given by $\phi(x, y) = (x, y, x^2 + y^2)$.

whether or not new data is likely to belong to the modelled class. The idea is as follows. Given a set of independent and identically distributed random variables $\{x_1, ..., x_n\}$ from a distribution P, find a subset S of the input space such that realizations of P have a predetermined probability to belong to S. This is done by constructing a function $h(x)$ which is positive on S and negative on its complement. The non-linear boundary is obtained by finding a transformation $\Phi(x) : X \to F$ from the original space X to a feature space F. Even if the data can not be separated using a linear boundary in X they might be linearly separable in F. This is illustrated for a simple 2-D case in Figure 2.4.

In the case of 1-class classification the objective function is slightly altered compared to the two class problem. Rather than finding a hyperplane separating two classes the goal is to find a hyperplane that separates all points from the origin and maximizes the distance from this plane to the origin. This gives the following expression [33]

$$\begin{array}{lll} \underset{\omega,\ \xi,\ \rho}{\min} & \frac{1}{2}\|\omega\|^2 + \frac{1}{\nu n}\sum_{i=1}^{n} \xi_i - \rho & \\ \text{s.t.} & (\omega \cdot \phi(x_i)) \geq \rho - \xi_i & \text{for all } i = 1, \ldots, n, \\ & \xi_i \geq 0 & \text{for all } i = 1, \ldots, n \end{array} \tag{2.17}$$

ν is a design parameter that can be used to shape the decision boundary and will be discussed later. Using the resulting hyperplane from Eq (2.17) the classifier in terms of primal variables can be expressed as:

$$h(x) = \text{sgn}((\omega \cdot \phi(x_i)) - \rho) \tag{2.18}$$

The new Lagrange function becomes

$$\mathcal{L}(\omega,\xi,\rho,\alpha,\beta)=\frac{1}{2}\|\omega\|^2+\frac{1}{\nu n}\sum_i \xi_i-\rho-\sum_i \alpha_i\Big((\omega\cdot\Phi(x_i))-\rho+\xi_i\Big)+\sum_i \beta_i\xi_i. \tag{2.19}$$

The maximum of this function can be located by examining $\nabla\mathcal{L}=0$, giving the following expressions

$$\frac{\partial L}{\partial\omega}=\omega-\sum_i \alpha_i\Phi(x_i)=0, \tag{2.20a}$$

$$\frac{\partial L}{\partial\xi}=\frac{1}{\nu n}\sum_i 1-\sum_i \alpha+\sum_i \beta=0, \tag{2.20b}$$

$$\frac{\delta L}{\delta\rho}=-1+\sum_i \alpha_i=0. \tag{2.20c}$$

This gives the following constraints

$$\omega=\sum_i \alpha_i\Phi(x_i) \tag{2.21}$$

$$\alpha_i=\frac{1}{\nu n}-\beta_i\le\frac{1}{\nu n} \tag{2.22}$$

$$\sum_i \alpha_i=1. \tag{2.23}$$

Substituting this into Eq. (2.19) the problem can be rewritten as

$$\begin{aligned}\min_{\alpha}\quad & \frac{1}{2}\sum_i\sum_j \alpha_i\alpha_j\Big(\Phi(x_i)\cdot\Phi(x_j)\Big)\\ \text{s.t.}\quad & 0\le\alpha_i\le\frac{1}{\nu n},\\ & \sum_i \alpha_i=1.\end{aligned} \tag{2.24}$$

This gives the alternate expression for the classifier

$$h(x)=\operatorname{sgn}\left(\sum_{i,j}\alpha_i\Big(\Phi(x_i)\cdot\Phi(x_j)\Big)-\rho\right). \tag{2.25}$$

It can be shown [34] that $\alpha,\beta\neq 0$ gives equality in the two inequality constraints of Eq. (2.17). The offset ρ can therefore be determined as

$$\rho = \big(\omega \cdot \Phi(x_i)\big) = \sum_{i,j} \alpha_j \big(\Phi(x_i) \cdot \Phi(x_j)\big) \tag{2.26}$$

Looking at Eq. (2.24) one can notice that solving this problem entails computing inner products a potentially high dimensional feature space F. This problem can be alleviated by applying something known as the kernel trick. The inner product can be replaced with a function $k(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$ called a kernel function. This allows for solving the optimization problem without having to perform an explicit projection $X \rightarrow F$ and computing all the inner product in that feature space. A popular choice of kernel is the radial basis function kernel or RBF kernel, described by

$$k(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{\tau}\right) \tag{2.27}$$

The behaviour of this kernel, and thus the decision boundary, can be influenced by the design parameter τ. Altering τ affects the "width" of the kernel. A small τ gives a narrow kernel meaning that only the closest points will affect the decision boundary which gives a jagged boundary. This potentially creates a lot of isolated "islands" and increases the risk of overfitting. If τ was very large on the other hand, the kernel would be extremely wide. The area of influence around each point would include a large portion of the other data and the decision boundary would be unable to capture the "shape" of the classes. The result is a very smooth boundary with a high risk of underfitting. The influence of τ is illustrated in Figure 2.5.

In the case 1-class SVM Eq (2.17), which is what is actually used in this study, the parameter ν is important for the the behaviour of the resulting classifier. It sets an upper bound on the outlier (training data treated out-of-class) fraction and, it is a lower bound on the number of training points used as support vectors. A small value of ν leads to a smaller number of support vectors and, therefore, a smooth, crude decision boundary. A large value of ν leads to a larger number of support vectors and, therefore, a curvy, jagged decision boundary. The optimal value of ν should be large enough to capture the data complexity and small enough to avoid overfitting.

2.3.2 Logistic regression

Logistic regression is a statistical method that can be used to model a binary dependant random variable, by estimating the parameters of a logistic model. A binary logistic model has a binary dependent variable Y, which denotes which class the sample belongs to, and a corresponding indicator variable X. The objective is to model the posterior probabilities $\Pr(Y = y|X = x)$ as a function which is linear in x [19].

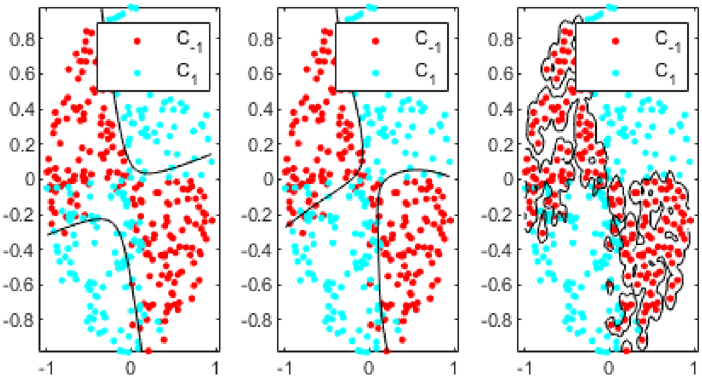


Figure 2.5: *Decision boundaries for SVM binary classifier using a Gaussian kernel for the same data using* $\tau = 5$ *(left)* $\tau = 1$ *(middle) and* $\tau = 0.05$ *(right).*

Let two classes be denoted Class C_1 and Class C_0, with the convention that $Y = 1$ if the sample belongs to C_1 and $Y = 0$ if the sample belongs to C_0. The goal is to model the posterior probability $\Pr(Y = 1|X = x)$. Assume this probability can be described as $\Pr(Y = 1|X = x) = p(x;\theta)$ for some function p and parameter θ. Note that $\Pr(Y = 0|X = x) = 1 - p(x;\theta)$. The conditional likelihood then becomes

$$\prod_{i=1}^{n} \Pr(Y = y_i|X = x_i) = \prod_{i=1}^{n} p(x_i;\theta)^{y_i}(1 - p(x_i;\theta)^{1-y_i}). \tag{2.28}$$

The idea is to find a parametrized model for p and then maximize the likelihood function with respect to θ. There are many possible choices for p but in the case of logistic regression the model is:

$$\log \frac{p(x)}{1 - p(x)} = \beta_0 + \beta^\intercal x, \tag{2.29}$$

which is linear in x, and where β_0 and β are model parameters. Here β is a parameter vector of the same length as x, where β_i is the weight x_i which is the i-th component of x. The parameter β_0 can be seen as a threshold separating the two classes. Solving this for p gives that

$$p(x;\beta,\beta_0) = \frac{1}{1 + e^{-(\beta_0 + \beta^\intercal x)}}. \tag{2.30}$$

Using the expression for the posterior probability obtained in Eq. (2.30) together with Eq. (2.28) gives the updated likelihood function $L(\beta, \beta_0)$ as:

$$L(\beta, \beta_0) = \prod_{i=1}^{n} \left(\frac{1}{1 + e^{-(\beta_0 + \beta^\intercal x)}} \right)^{y_i} \left(1 - \left(\frac{1}{1 + e^{-(\beta_0 + \beta^\intercal x)}} \right)^{1-y_i} \right) \tag{2.31}$$

It is often more convenient to work with the log-likelihood function $l(\theta) = \log(L(\theta))$ rather than the likelihood function. Using Eq. (2.31); $l(\theta)$ is given by

$$l(\beta, \beta_0) = \sum_{i=1}^{n} \left[y_i \cdot (\beta_0 + \beta^\intercal x_i) - \log(1 + e^{\beta_0 + \beta^\intercal x}) \right]. \tag{2.32}$$

The maximum log-likelihood estimation of β and β_0 is thus given as

$$\underset{\beta, \beta_0}{\arg\max} \quad \sum_{i=1}^{n} \left[y_i \cdot (\beta_0 + \beta^\intercal x_i) - \log(1 + e^{\beta_0 + \beta^\intercal x}) \right]. \tag{2.33}$$

This is a convex problem [19], which can be maximized using e.g. the Newton-Raphson method as explained in [25].

Logistic regression can also be used as a feature selection method. One way to reduce the amount of parameters used to represent the data is to only consider explanatory variables corresponding to $\beta_i \neq 0$. As explained in Section 1.1.2, the goal is often to generate a low dimensional data representation which implies that the parameter vector β should be sparse. This can be enforced by adding an L_1 penalty to Eq. (2.33) giving the following expression:

$$\max_{\beta_0, \beta} \quad \left\{ \sum_{i=1}^{n} [y_i \cdot (\beta_0 + \beta^\intercal x_i) - \Psi(\beta_0, \beta, x_i)] - \lambda \sum_{j=1}^{p} |\beta_j| \right\}. \tag{2.34}$$

where $\Psi(\beta_0, \beta, x_i) = \log(1 + e^{1 + \beta_0 + \beta^\intercal x_i})$, and λ is used to regulate the sparsity of the solution. Increasing λ reduces the number of elements $\beta_i \neq 0$ and $\lambda = 0$ removes the penalty and gives Eq. (2.33).

3 Proposed algorithm

Model-based and data-driven approaches are two categories of diagnostic methods used in fault detection and diagnosis (FDD). Model-based methods require extensive knowledge about a supervised system to have the ability to derive models accurate enough to detect deviations from nominal behaviour. This is often time consuming and might not even be feasible for large, complex systems. Data-driven approaches on the other hand are not significantly model dependent, but rely heavily on the availability of run-time data. In the case of supervised learning, the training data also needs to be labelled, meaning that all data has to be assigned a class e.g. it is known what kind of fault mode, including the fault free system, the data is collected from. There are solutions using hybrid approaches of these methods where information about the physical system facilitate the data analysis or conversely, shortcomings of the model can be compensated for by a data-based method. This study belongs to the last category where a data-driven classifier is designed on top of a model-based framework. This chapter explains how such a classifier is created.

Given the objective stated in Section 1.3, the following requirements are set for the classifier. Given new data, the classifier should be able to

1. Detect any faults in the system.
2. Determine if data is consistent with a known fault, and if so which.
3. Estimate the size/severity of the fault.

3.1 Residual selection

The first step of the classifier design is to detect if the behaviour deviates from the nominal case i.e., there is a fault present in the system. By utilizing information about the physical properties of the system, a set of residual generators R is constructed. How this is done is outside the scope of this study and for further details, see for example [10]. If the model is perfect and it is known which residuals are sensitive to which fault mode, fault detection would simply be a matter of picking the set R_i that is sensitive to each fault f_i and say that fault f_i is present in the system if the output from this residual set is non-zero. Since the models are never perfect in reality, the problem is rarely this simple. The algorithm proposed here demonstrates a method for residual selection when the model is imperfect, without making any assumptions about the sensitivity of the residual generators.

This is essentially a feature selection problem and as such there are a variety of approaches available. An introduction to this field is found in [15] or [4]. For this method, L1-logistic regression (described in Section 2.3.2) is used. Let $\bar{r}[t]$ be a column vector of the output from the residual generator set at time $t \in [0, N]$ and be normalized in the nominal case (NF). The data is labelled so that for each time instance, the class of the residual vector $\bar{r}[t]$ is known $\left[\{\bar{r}[0], y_0\} \ldots \{\bar{r}[N], y_N\}\right]$. Introduce a function $c[t]$ such that $c[t] = 1$ when $\bar{r}[t]$ belongs to class 1 and $c[t] = 0$ when $\bar{r}[t]$ belongs to class 0. The goal is to create an affine function $\beta_0 + \beta^\intercal \bar{r}[t]$ such that $\beta_0 + \beta^\intercal \bar{r}[t] \geq 0$ when $\bar{r}[t]$ belongs to class 0 and $\beta_0 + \beta^\intercal \bar{r}[t] < 0$ when $\bar{r}[t]$ belongs to class 1. Using Eq. (2.34), β and β_0 are chosen as the solution to

$$\underset{\beta_0, \beta}{\arg\max} \left\{ \sum_{t=1}^{N} \left[c[t] \left(\beta_0 + \beta^\intercal \bar{r}[t] \right) - \Psi(\beta_0, \beta, \bar{r}[t]) \right] - \lambda |\beta_j| \right\}. \tag{3.1}$$

where $\Psi(\beta_0, \beta, \bar{r}[t]) = \ln(1 + e^{\beta_0 + \beta^\intercal \bar{r}[t]})$. The parameter λ is of central importance in the design process since it affects the cardinality of the residual generator set $|R_i|$. Assume that λ is chosen such that $|R_i| = k$ for some $k \leq M$ where $M = |R|$. In this case, the data D on which the classifier T_i is used, would be represented in a k-dimensional space $T_i : D \longrightarrow \mathbb{R}^k$. The curse of dimensionality states that: as k grows so does the numerical complexity of the classification and the amount of data required to maintain observation density. In the residual selection process, Eq. (3.1) is solved for different values of λ and the performance of each candidate set is evaluated using cross-validation [27]. After this, R_i is chosen as "the smallest set with acceptable performance". Exactly how λ should be chosen is non-trivial and one method is by using the method provided by the GLMnet toolbox for Matlab [29].

For the classifier design, residuals are selected for each fault type separately so that one classifier is T_i is constructed for each fault mode f_i. This is done by solving Eq. (2.34) where class 0 = NF and class 1 = f_i, for each f_i. This assures that all faults are detectable, assuming that for each f_i there is a set of residual generators $R_i \neq \emptyset$ sensitive to f_i.

The goal of the residual selection step is shown in Figure 3.1. In this 2-dimensional example, residuals r_a & r_b are selected since they together create a space in

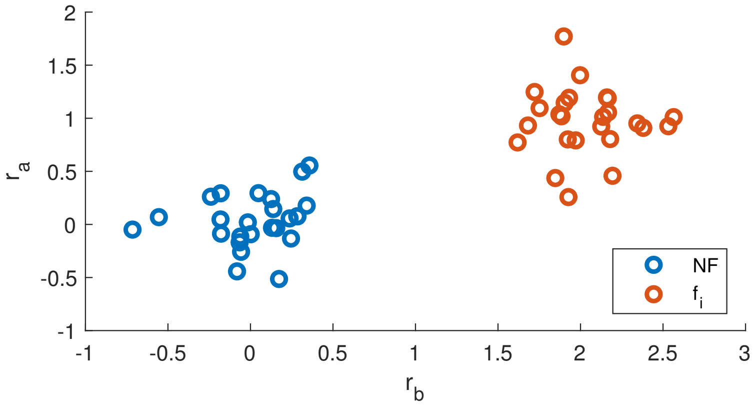


Figure 3.1: *Residuals r_a and r_b are selected to achieve maximum separation between the classes* NF *and f_i.*

which the data from fault mode f_i is clearly separated from the nominal data.

3.2 Data processing

Part of the purpose of this study is to reduce the amount of data needed in the diagnostic framework. This is partly accomplished by the residual selection since $|\bigcup_i R_i| \leq |R|$. The residual selection step is not guaranteed to reduce the number of used residuals depending of the redundancy of the available residuals. In the worst case, the classifier will use all available residuals meaning that the number of signals to monitor will not decrease at all.

The next step of the data compression is to reduce the information needed to represent each residual. This is done by segmenting and parametrizing the residual in each of the classifier subspaces. Let $r_j(z[t])$ be residual j, sampled at time t, and $z[t]$ be a set of actuator inputs and/or system output signals. The notation $r_j[t] = r_j(z[t])$ is used for convenience. Partition the training data from each residual r_j into segments of length l so that the n-th segment can be expressed as

$$r_{j,n} = r_j([(n-1)\cdot l, n\cdot l]), \quad n = 1, ..., \lfloor t_{fin}/l \rfloor. \tag{3.2}$$

The segmentation is illustrated in Figure 3.2. This shows the partitioning of a 1-dimensional residual generated from a 1800 s long measurement cycle. The residual is partitioned into 30, 60 s segments and the vertical lines indicates at which points the residual is split.

To construct the parametric representation of each data segment, a Gaussian probability distribution estimation is performed in each classifier subspace. Let r^i be the residuals generated by R_i. The vector is approximated as a k-dimensional Gaussian distribution $\mathcal{N}^k$, where $k = |R_i|$. This is written as

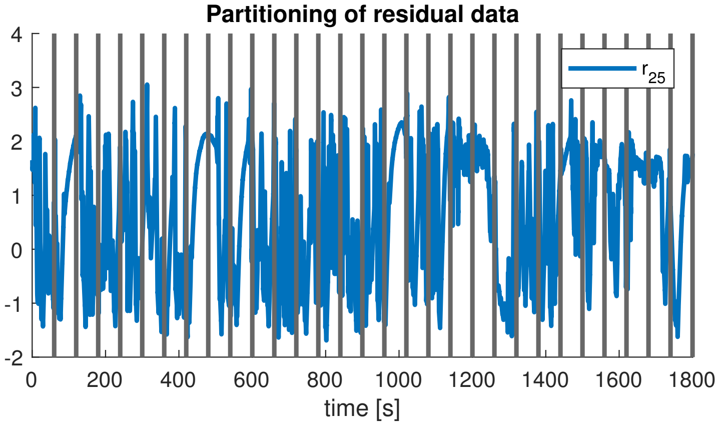


Figure 3.2: *One residual r_{25}, being partitioned into segments of length l.*

$$r^i_{j,n} \dot{\sim} \mathcal{N}(\hat{\mu}, \hat{\Sigma}), \tag{3.3}$$

where $\hat{\mu}$ and $\hat{\Sigma}$ are the sample mean vector and sample covariance matrix of $r^i_{j,n}$. This process is illustrated for a single residual in Figure 3.3. This illustrates how 1-dimensional Gaussian distributions are fitted to three 60 s segments of the residual. The distributions are drawn along the vertical lines separating the segments and their amplitude has been scaled for the sake of visibility.

The compression ratio is defined as the ration between the size of the uncompressed data and the compressed data [32]. To examine what effect this approximation has on the data storage, a rough estimate of the compression ratio Δ is made as follows. Let the length of each residual vector be N samples. To represent this data in a residual subspace of dimension k would require $N \cdot k$ values to be stored. Partition the data into n segments, where each segment is represented by a $k \times 1$ mean vector and a $k \times k$ covariance matrix. The size of one segment is $k + k^2$ and the total number of segments is N/n. The compression ratio can then be expressed as:

$$\Delta = \frac{N \cdot k}{\frac{1}{n} N \cdot k(1+k)} = \frac{n}{1+k}. \tag{3.4}$$

Say for example that a system is monitored by sensors measuring with a frequency of 100 Hz, and that the segment length is 30 s giving $n = 30 * 100 = 3000$. Assume that two residuals are chosen for the classifier ($k = 2$). This gives $\Delta = 3000/3 = 1000$, meaning that the original data is 1000 times larger than the compressed data.

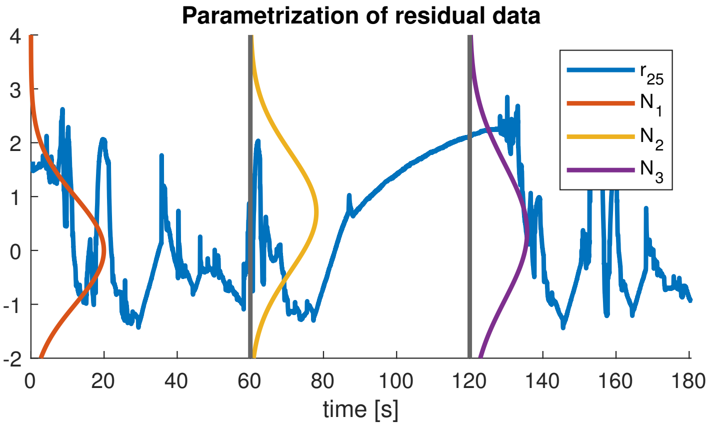


Figure 3.3: *Illustration of the parametrization of residual segments when ($r \in \mathbb{R}^1$). The distributions have been scaled for illustrative purposes.*

3.3 Data classification

After the residual sets are selected for each fault, the next step is for the classifier to distinguish between data from different fault modes, essentially a classification problem. This is solved by using 1-class support vector machines (SVM) to model the different fault modes. An explanation of SVM is given in Section. 2.3.1. Consider a classifier T_i. The training data is represented as distributions $\bar{q}$. Associated with each distribution q_j is a categorical variable $y_j \in F$, where F is the set of all known fault modes. A 1-class SVM is constructed by solving the following optimization for each fault mode:

$$\begin{aligned} \min_{w,\ \xi,\ \rho} \quad & \frac{1}{2}\|\omega\|^2 + \frac{1}{\nu n}\sum_{j=1}^{n}\xi_j - \rho \\ \text{s.t.} \quad & (\omega \cdot \phi(\bar{q}_i)) \geq \rho - \xi_j \quad \text{for all } j = 1, \ldots, n, \\ & \xi_j \geq 0 \quad \text{for all } j = 1, \ldots, n, \end{aligned} \tag{3.5}$$

where $\bar{q}_i$ is the set of distributions corresponding to $y_j = f_i$. This gives the classifier

$$h_i(x) = \operatorname{sgn}(\omega_i \cdot \phi(x_i) - \rho_i) \tag{3.6}$$

where ω_i, ρ_i are the solutions to Eq. (3.5) for fault mode f_i. Figure 3.4 illustrates a 1-class SVM classifier. The algorithm creates a boundary $h_i(x) = 0$ which encapsulates the data from the example class while maintaining a "tightness" of the boundary. 1-class SVM has two main advantages over multi-class SVM in the case of diagnostic. The first is that it allows for overlapping classes. Data can si-

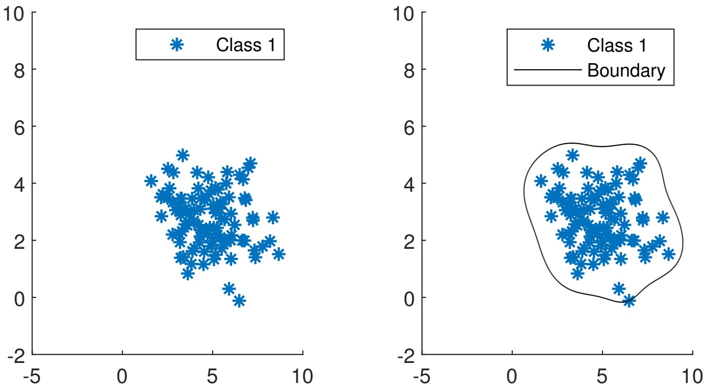


Figure 3.4: *Illustration of a 1-class SVM classifier for a class "Class 1". The decision boundary is given by $h(x) = 0$.*

multaneously lay inside several classes at once. A physical interpretation of this property is that there are several possible explanation for the observed data and that the true diagnosis is lost due to e.g. unbalanced data sets. The other advantage is the the 1-class SVM allows for data not to be part of any known class and as such provides a convenient method of novelty detection.

The parameter ν and transform function $\Phi(\cdot)$ are discussed in closer detail in Section. 2.3.1, but it is important to note that both of these are considered design parameters and can be used to affect the shape of the hyperplane $\omega_i \cdot \phi(x_i) - \rho_i = 0$. These parameters adjust the trade-off between the risk of overfitting and the missed-detection rate.

3.3.1 Partial- and final diagnosis

Classification of new observations p is done by evaluating each class separately. If $h_i(p) > 0$, fault mode f_i considered a possible explanation of p. Let $T_i(p) = \bigcup f_i$ s.t. $h_i(p) > 0$. The partial diagnosis d is the intersection of all fault modes consistent with each classifier:

$$d(p) = \bigcap_{i \in F} T_i(p). \tag{3.7}$$

There are two cases that are of special interest concerning the diagnosis. When NF $\in d$, all of the classifiers were unable to tell if the data deviated from the fault free training data. In this case, the system is said to be fault free. Note that this does not guarantee that the system is fault free, it only says that, given historical data and current observations, there is no reason to suspect that the system is malfunctioning.

The other interesting case is when $d = \emptyset$. In this case, none of the fault mode models were able to explain the new data. This could possibly be because the new data is a realisation of a known fault type with a fault severity that is sufficiently different from any training data that it is not captured by the dedicated SVM. Another possibility is that the data is a realisation of a fault mode which is not present in the training data, and as such, is said to be an unknown fault.

This could be summarized as follows. The final diagnosis D is a function of the partial diagnosis $d(p)$ such that

$$D\big(d(p)\big) = \begin{cases} \text{NF}, & \text{if NF} \in d, \\ \text{Unknown}, & \text{if } d = \emptyset, \\ d, & \text{otherwise.} \end{cases} \tag{3.8}$$

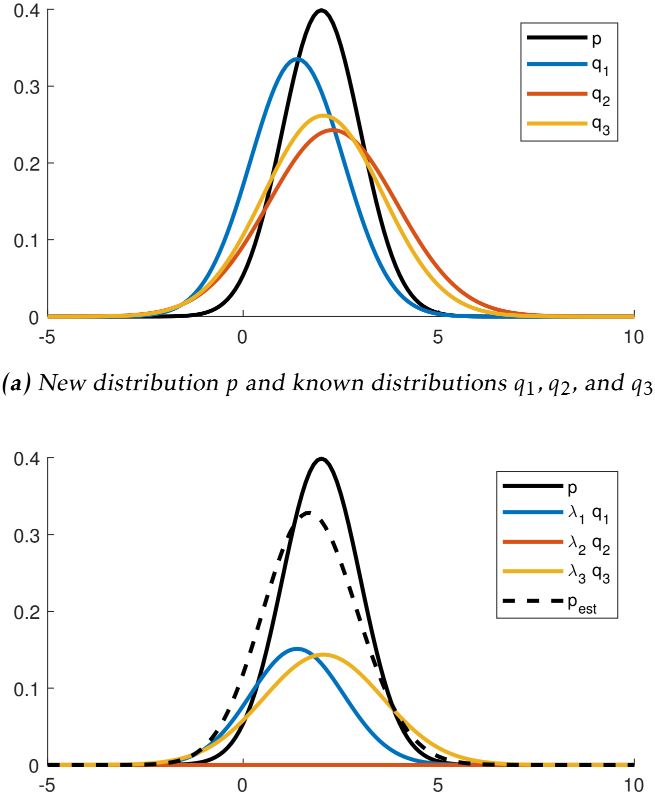
3.4 Fault severity estimation

The method presented in Section 3.3 provides a means to classify new data, but it does not give any information about the severity of these faults. If each training distribution q_i has a known fault size θ_i, this information can be utilized to estimate the severity of new faults by comparing how similar the data is to the training data. One approach that has been suggested [12] is to model faults into qualitative classes, such as {normal, slight, large}. Another way, which is a method that is largely unexplored, is to find a quantitative severity estimation $\hat{\theta}$. This study suggests a method for estimating $\hat{\theta}$ as a convex optimization problem by using the Kullback-Leibler divergence (KLD) as a dissimilarity measure. The method is based on the following fundamental assumption.

Assumption 3.1. New data, collected from a fault mode f_i of severity θ_i, with distribution p_i should be "close" (in a KLD-sense) to training data from the same fault mode and severity in the residual space.

The KLD is introduced in Section 2.1. There is a variety of different statistical divergence measures, see [1] or [2] for some examples. The reason for using the KLD is that is has been shown to be a powerful tool to detect slight changes in the data distribution, and has been used in e.g. [11][16] [41] [42] [43].

The first step of the severity estimation is using the training data to characterize the input data. Let p denote the Gaussian approximation of the distribution of the input data and let θ denote the severity of this fault. Reconstruct p as a linear combination of training distributions $q_1, \dots, q_k$, such that the estimated distribution $\hat{p}$ minimizes $\mathcal{K}(p|\hat{p})$ where $\hat{p} = \sum_{i=1}^{k} \lambda_i q_i$. This can be summarized as:



***(a)** New distribution p and known distributions q_1, q_2, and q_3*

***(b)** Reconstructed $\hat{p}$ where $\lambda = \{0.45, 0, 0.55\}$.*

***Figure 3.5:** Illustration of the parameter estimation process.*

$$\begin{aligned}
\underset{\lambda_1 \ldots \lambda_k}{\arg\min} \quad & \mathcal{K}\Big(p(x) \| \lambda_1 q_1(x) + \lambda_2 q_2(x) + \ldots + \lambda_k q_k(x)\Big), \\
\text{s.t.} \quad & \sum_{i=1}^{k} \lambda_i = 1, \\
& \lambda_i \geq 0, \quad \forall i.
\end{aligned} \tag{3.9}$$

where the condition $\sum_{i=1}^{k} \lambda_i = 1$ is added to normalize the solution. Figure 3.5 shows a 1-dimensional example of this procedure. Training distributions $q_{1,\ldots,3}$ are used to classify an unknown distribution p. The original distributions are shown as well as the estimated distribution $\hat{p}$ along with the weighted distributions $\lambda_{1,\ldots,3} \cdot q_{1,\ldots,3}$. In this example, only two of the training distributions have non-zero weights i.e. are used for the estimate.

Let λ^* denote the solution to Eq. (3.9) and let λ_i^* be the i-th element of λ^*. The fault severity is then estimated as

$$\hat{\theta} = \sum_{i=1}^{k} \lambda_i^* \theta_i \tag{3.10}$$

where θ_i is the fault size of q_i in Eq.(3.9). To evaluate the target function in Eq. (3.9), the KLD between a Gaussian distribution and a Gaussian mixture model has to be computed. As stated in Section 2.1, this problem has no closed form solution and by using the Monte Carlo approximation given by Eq.(2.3) together with Eq. (3.9), the updated problem becomes:

$$\begin{aligned} \underset{\lambda_1 \ldots \lambda_k}{\arg\min} \quad & \frac{1}{n} \sum_{1=1}^{n} \ln \left(\frac{p(x_i)}{\lambda_1 q_1(x_i) + \lambda_2 q_2(x_i) + \ldots + \lambda_k q_k(x_i)} \right) \\ \text{s.t.} \quad & \sum_{i=1}^{k} \lambda_i = 1, \\ & \lambda_i \geq 0, \quad \forall i. \end{aligned} \tag{3.11}$$

A non trivial question is how the reference data set $Q_k = \{q_1, \ldots, q_k\}$ should be chosen from the available training data. This is a two part problem of how to choose k and which reference distributions q_i to select.

One approach would be to simply include all distributions from the given fault e.g. $k = N_i$ where N_i is the number of realisations of fault f_i. This is an ineffective method since it increases the computational cost by adding numerous distribution q_j likely to correspond to $\lambda_j = 0$. If data has been collected from a variety of severities and conditions, it is unlikely that new data would have a distribution that is equally similar to all available training data. Using this line of reasoning, only a small subset of all realisations are reasonably of interest for $\hat{p}$.

Here, Q_k is created using a version of k-nearest neighbour (k-NN) selection. The selection is defined as

$$\begin{aligned} \underset{Q_k}{\arg\min} \quad & \sum_{q_i \in Q_k} \mathcal{K}(p|q_i), \\ \text{s.t.} \quad & Q_k \subseteq Q, \\ & |Q_k| = k, \end{aligned} \tag{3.12}$$

where the sum is taken over all elements in Q_k. Since $\mathcal{K}(p|q_i) \geq 0, \forall i$, this will give the set of the k distributions closest to p in a KLD -sense. This means that k is a design parameter to adjust the trade-off between performance and computational cost. Figure 3.6 shows an example of the k-NN selection using $k = 10$ for an

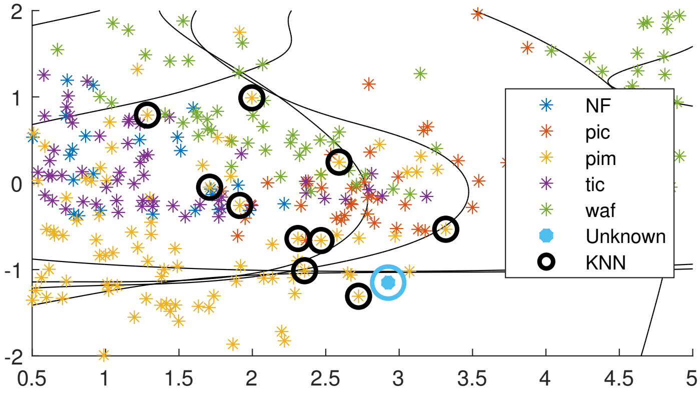


Figure 3.6: *The k-nearest neighbours used for severity estimation, with the new distribution p denoted "Unknown".*

unknown fault from fault mode $f_{p_{im}}$. Note that the selected distributions are not the closest to p in an euclidean sense. The markers only show the mean of the involved distributions and the KLD based k-NN method also accounts for the covariance of the distributions.

The problem formulation in Eq. (3.12) is convenient since both q and q_i are Gaussian distributions for all $i = 1, \ldots, N_i$ meaning that Eq. (2.2) can be used to calculate the KLD.

3.5 Full algorithm

The full algorithm can be separated into two parts; training and classification.

The training is done through the following steps:

1. Collect training data and generate residuals where each sample is labelled with fault mode f_i and severity θ.

2. Use L1-logistic regression to generate a residual set R_i for each f_i.

3. Partition and parametrize the residual training data in each space R_i.

4. For each R_i create a 1-class support vector machine to model each fault mode individually.

Once the classifier is trained, classification of new residual data is performed as follows:

1. Partition and parametrize the new data in each subspace R_i.

2. For each classifier and each fault mode, use the 1-class SVM to check if the new data is consistent with the fault modes represented in training data.

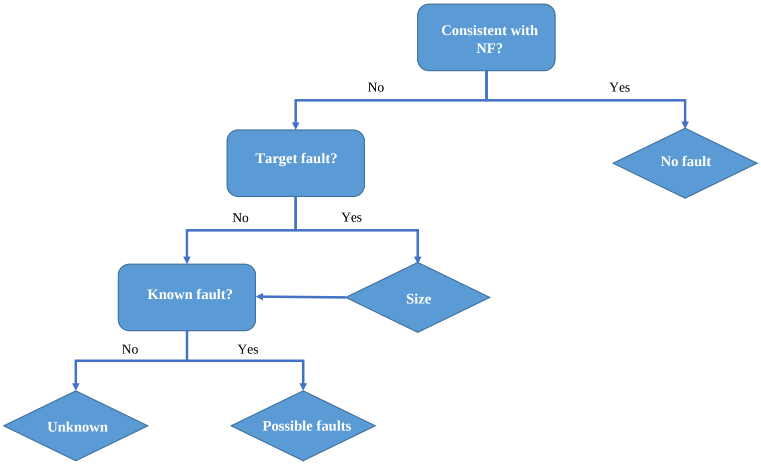


Figure 3.7: *Overview of the full algorithm.*

3. For each classifier, if p is distinguishable from NF and consistent with f_i, where f_i is the target fault of T_i, estimate the severity $\hat{\theta}$ using the optimization presented in Section 3.4.

4. Construct the final diagnosis as the intersection of the partial diagnoses, using the interpretation as shown in Eq. (3.8).

The decision structure used in the classification processes is visualized in Figure 3.7. The schematic shows how new data is evaluated in each classifier to generate the partial diagnosis.

4 Results

The diagnostic framework is tested by using experimental data collected from an engine test bench. The engine is a commercial, turbo charged, four cylinder, internal combustion engine from Volvo, and the test bench in question is seen in Figure 4.1. The sensors and actuators used are the standard commercial configuration for the engine.

4.1 Data collection

Data was collected from four different sensor faults and from the fault free system. The faults are introduced by altering the sensor output gain in the engine control system. The nominal output $z[t]$ is multiplied by a factor θ so that the resulting output $\tilde{z}[t]$ is given as:

$$\tilde{z}[t] = \theta \cdot z[t]. \tag{4.1}$$

The notation $\{f_{p_{ic}}, +20\%\}$ indicates a fault of type p_{ic} (senor measuring the intercooler pressure) and severity $\theta = 1.2$. A list of all the fault modes, along with their respective severities, used in the data collection is found in Table 4.1.

The residuals are generated by the Matlab Fault Diagnosis toolbox [10] using an existing model, similar to the one described in [8], which models the flow of air through the engine. A schematic view of the engine along with the monitored parameters is shown in Figure 4.2 were y are sensor measurements and u are actuator signals.

The data was generated using the class 3 Worldwide harmonized Light-duty vehicles Test Cycles (WLTC), which is part of the World harmonized Light-duty ve-

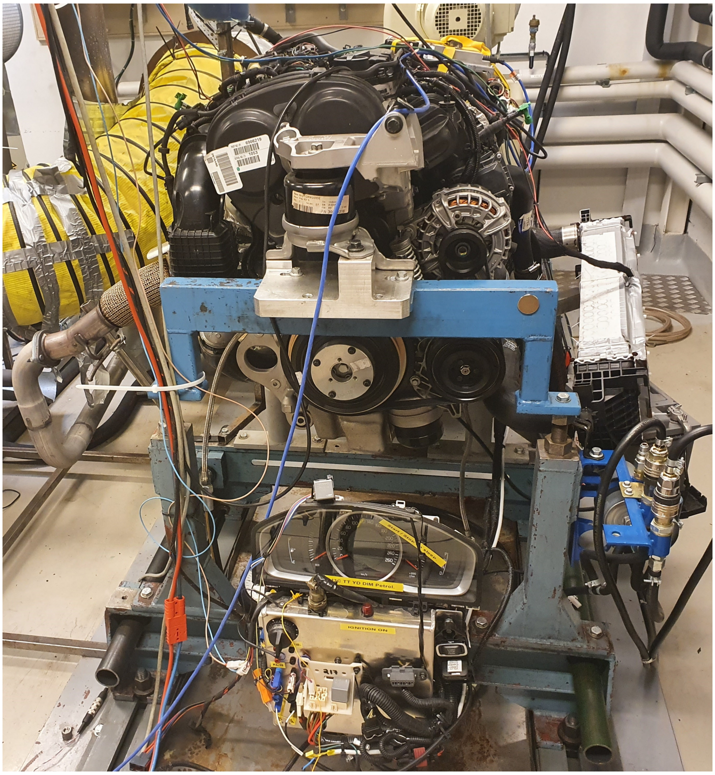


Figure 4.1: The engine test bench used to collect experimental data.

Table 4.1: *Data was collected from four different sensor faults. Faults were introduced by altering the gain on the sensor output.*

Notation	**Type of sensor fault**	**Severities** [%]
$f_{T_{ic}}$	Intercooler temperature	-20, -10, 10, 20
$f_{p_{ic}}$	Intercooler pressure	-20, 10, 20
$f_{p_{im}}$	Intake manifold pressure	-20, -10, 10, 20
$f_{W_{af}}$	Wastegate air flow	-20, -10, 20
NF	Fault free	-

hicles Test Procedure (WLTP). The cycle is explained in detail in [36], and the speed profile of the cycle is shown in Figure 4.3. The cycle is used since it covers a variety of operating point where the load can be split into four different parts of the cycle: *low, medium, high, very high*. The benefit of collecting data from different operating points is to account for any variance in the model output error due to varying speed and load. One example of why it is reasonable to assume that the model error is operating point dependent is the pressure measurements. Consider an air leakage in the intake manifold. The air mass flow through the

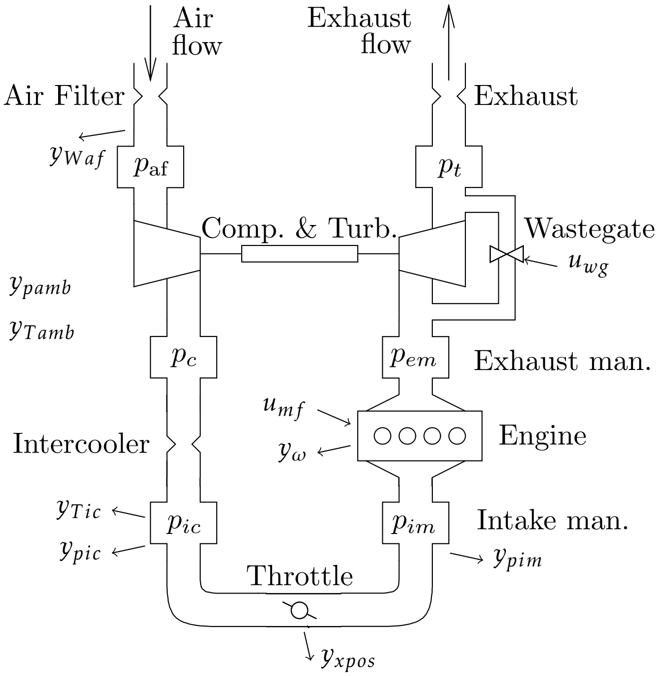


Figure 4.2: *A schematic of the model of the air flow through the model. Available output signals are sensors y and actuators u. The plot is used with permission from [9].*

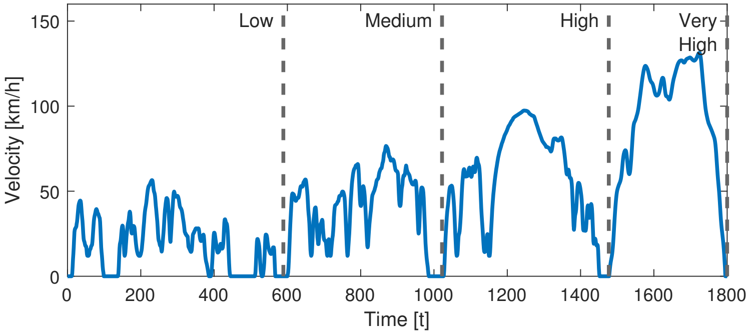


Figure 4.3: *Speed profile for the class 3 worldwide harmonized light vehicles test cycle.*

hole would depend on the difference in pressure between the manifold and its surroundings, which at a *very high* operations would be greater than at *low* operations, and thus have a greater effect on the system.

Data is collected from the faulty and nominal system. An example of output data is shown in Figure 4.4, which shows a comparison between the intercooler pressure sensor in fault free mode and with a +20 % fault. The first 120 s of the $\{f_{p_{ic}}, +20\%\}$ data is collected from the nominal system meaning that the two signals are almost identical. After that interval, the figure shows a clear change in signal behaviour but that there is still overlap in the two signals, even when the severity is as big as 20 %.

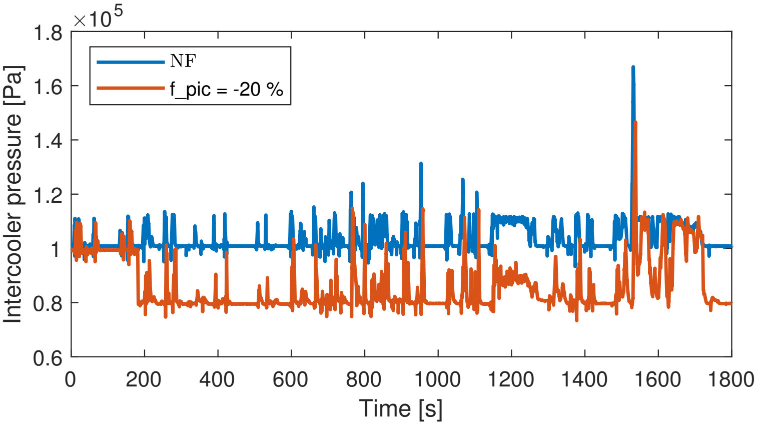


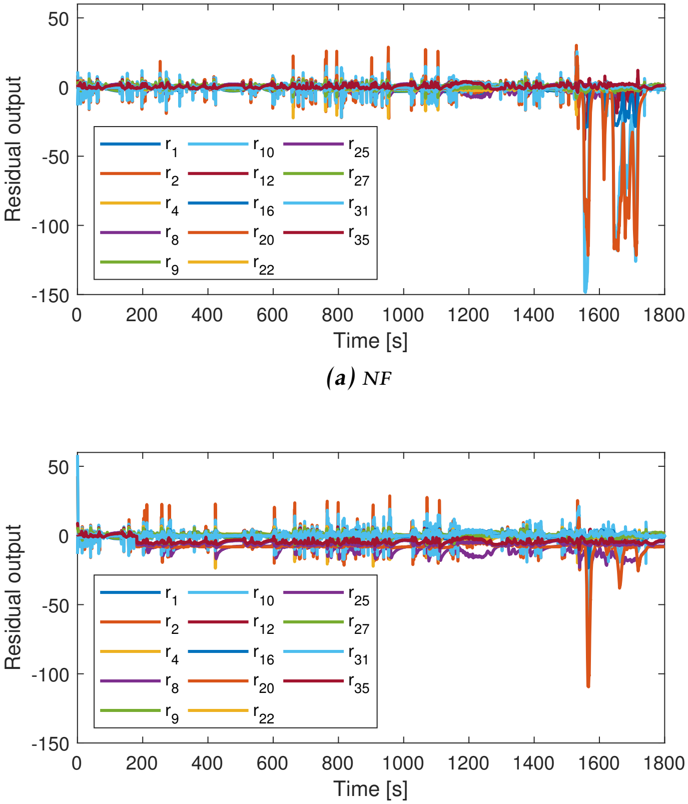
Figure 4.4: *Intercooler pressure sensor from* NF *and* $f_{p_{ic}}$ $- 20\,\%$. *The first two minutes are used for calibration and is fault free in both sets.*

The sensor and actuator data was re-sampled to each have a 100 Hz sample frequency and parametrized using a segment length of 30 s. Using Eq. (3.4), this gives a compression ratio $\Delta = 3000/(2+1) = 1000$ for $k = 2$. This says that the residual data is 1000 times larger than the parametrized version. In terms of storage this could mean using storage in the order of megabyte rather than gigabyte required for diagnosis.

4.2 Classifier training

The Matlab Fault Diagnosis toolbox [10] was used to generate a set of residual generator candidates, which gave a set of 35 candidate residuals. These residuals are not guaranteed to be stable and diverging residuals had to be removed manually. This left 14 residuals that are stable on all the training data sets.

Since the data was collected at different times, the model bias may vary between measurements as the error in initial states of the model and engine condition were different. To account for this, the data was collected so that the first 120 seconds of each data set is fault free, a part which is used to calibrate the generated residuals. The calibration was done so that the residuals are normalized to have zero mean and unit variance in their respective calibration set. Figure 4.5 shows the normalized residual output from two different tests: NF and $\{f_{p_{ic}}, -20\%\}$. The figure shows that the model is indeed operating point dependent as seen in the fault free set. The behaviour at the *very high*-part at the end of the cycle differs considerably from operations at lower loads. This figure also shows that a traditional methods of using residual thresholds (fault is said to be present if $r > c$ and absent if $r \leq c$ for some constant threshold c) are ineffective since the variance within the data is larger than the variance between data sets. This means that choosing a constant c, such that the classifier would have low rates of both false alarm and missed detection, is impossible.

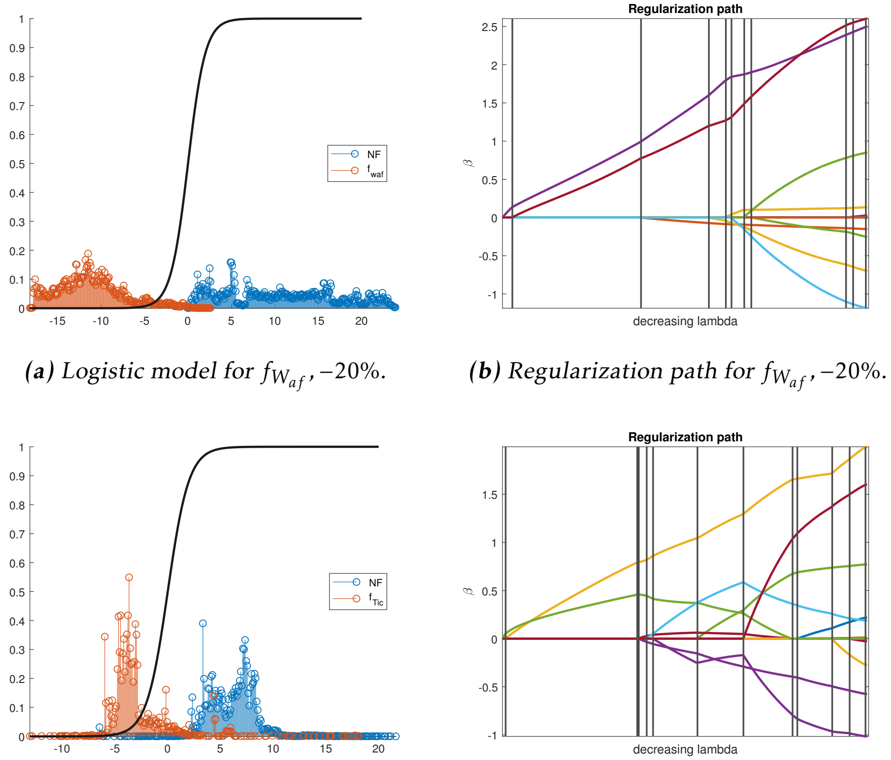


(a) NF

(b) $f_{p_{ic}}$, −20%

Figure 4.5: *Normalized output from the stable residuals in cases* NF *and −20% intercooler pressure sensor gain.*

After normalization, classifiers were created by selecting residuals using L1-regularized regression as explained in Section 3.1. The factor which determines which residuals are selected for a specific fault is the parameter λ in Eq. (3.1). This can be used to examine the regularization path of the coefficient vector β, in the same expression, by solving the problem for different values of λ. A systematic way to examine how β depends on λ is by using the GLMnet toolbox in Matlab [29]. Two examples of the regularization are presented in Figure 4.6, where the logistic model and regularization path for fault modes $\{f_{w_{af}}, -20\%\}$ and $\{f_{T_{ic}}, +20\%\}$ are shown. The left figures displays the resulting logistic model and the right figures illustrate the regularization paths of β. The regularization figure shows how the parameter vector β changes with λ. The left side correspond to the biggest penalty λ and the right side corresponds to the smallest λ. The vertical lines indicate all values of λ for which any element β_j of β goes from zero to non-zero or vice versa. Between the lines the elements of the solution are constant in that interval. The cardinality of the solution (number of non-zero elements in β) generally increases as λ decreases but there are instances at which a decrease in λ reduces the cardinality. This depends on the penalty cost, as an increase in the absolute value of one parameter β_j causes another parameter β_l to go from non-zero to zero and thus reduce the cardinality of the solution. Note that as λ decreases, the risk of overfitting increases, regardless of solution cardinality.



***(a)** Logistic model for $f_{W_{af}}, -20\%$.*

***(b)** Regularization path for $f_{W_{af}}, -20\%$.*

***(c)** Logistic model for $f_{T_{ic}}, 20\%$.*

***(d)** Regularization path for $f_{T_{ic}}, 20\%$.*

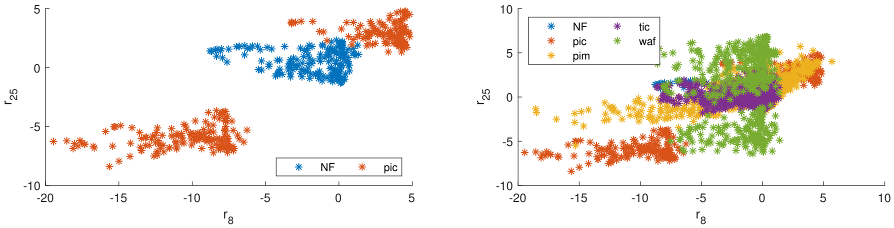
***Figure 4.6:** Illustration of the logistic model and regularization path of β for fault modes: $f_{W_{af}}, -20\%$ and $f_{T_{ic}}, 20\%$. The vertical lines in the regularization plots mark when the number of non-zero elements in β changes. Each interval corresponds to one subset of the residual candidates.*

Cardinality is generally a trade off between accuracy and overfitting but in this case it is also a matter of dimensionality, meaning that low cardinality solutions are preferred. Which solution that is used can thus be considered a design parameter. In this study the highest accuracy set with cardinality 2 was chosen to facilitate graphical illustration of the rest of the process. The residual sets selected for each fault mode is shown in Table 4.2.

Table 4.2: *Residuals selected for each fault mode using L1-logistic regression. The two best residuals from each fault mode are used for diagnosis.*

Fault mode	Sensor	Selected residuals
$f_{T_{ic}}$	Intercooler temperature	r_{17}, r_{22}
$f_{p_{ic}}$	Intercooler pressure	r_8, r_{25}
$f_{p_{im}}$	Intake manifold pressure	r_{25}, r_{31}
$f_{W_{af}}$	Wastegate air flow	r_{25}, r_{35}

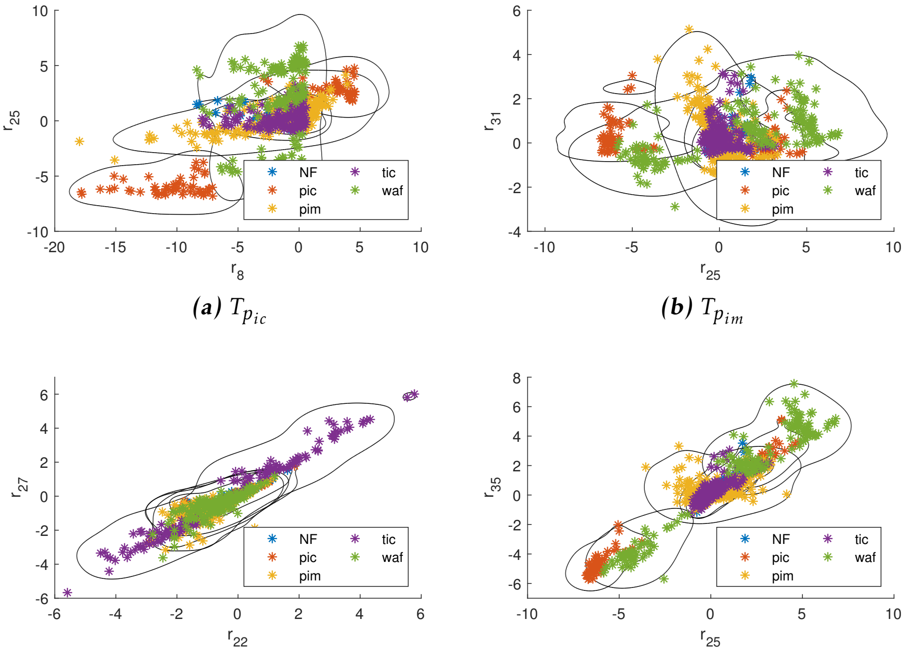
A classifier was then constructed for each fault mode using the residuals from Table 4.2, giving the four classifiers $\{T_{T_{ic}}, T_{p_{ic}}, T_{p_{im}}, T_{W_{af}}\}$, where each classifier is designed to detect a specific fault. Figure 4.7 shows how the means of the training data distributions from classes NF and $f_{p_{ic}}$ (which is the target fault) are distributed in the $T_{p_{ic}}$ classifier space. The data from the target fault and the nominal data are distinctly separated, and fault realizations with $\theta < 1$ and $\theta > 1$ are shown to the lower left and upper right side of the figure respectively. This indicates that the classifier is indeed sensitive to $f_{p_{ic}}$, as intended with the residual selection. It is also clear that, even though the classifier is designed to detect $f_{p_{ic}}$, it is still sensitive to other fault modes. This is to be expected since the same residuals are used in more than one of the classifiers as shown in Table 4.2. Residual r_{25} were used for two other faults besides $f_{p_{ic}}$. If a residual is chosen to detect a fault f_i it should still be sensitive to that fault if it is selected to detect another fault f_j as well.



(a) *Only* NF *and target fault $f_{p_{ic}}$.* ***(b)*** *All faults from training data.*

Figure 4.7: *Illustration of residual representation in the $T_{p_{ic}}$ classifier subspace. The classifier separates the target fault ($f_{p_{ic}}$) from* NF*, but is also sensitive to other faults in the training data.*

Once the residuals are selected, a 1-class SVM is created for each fault mode, using $\nu = 0.5$ and the radial basis function as kernel. All severities of a single fault type is modelled as a single class, representing that specific fault mode. The decision boundaries for each SVM are shown in Figure 4.8 for the four classifiers. The figure shows significant overlap between classes in all off the classifier spaces meaning that part of the residual space belongs to multiple classes and will thus yield multiple possible faults in the partial diagnosis for realizations in that area.



(a) $T_{p_{ic}}$

(b) $T_{p_{im}}$

(c) $T_{t_{ic}}$

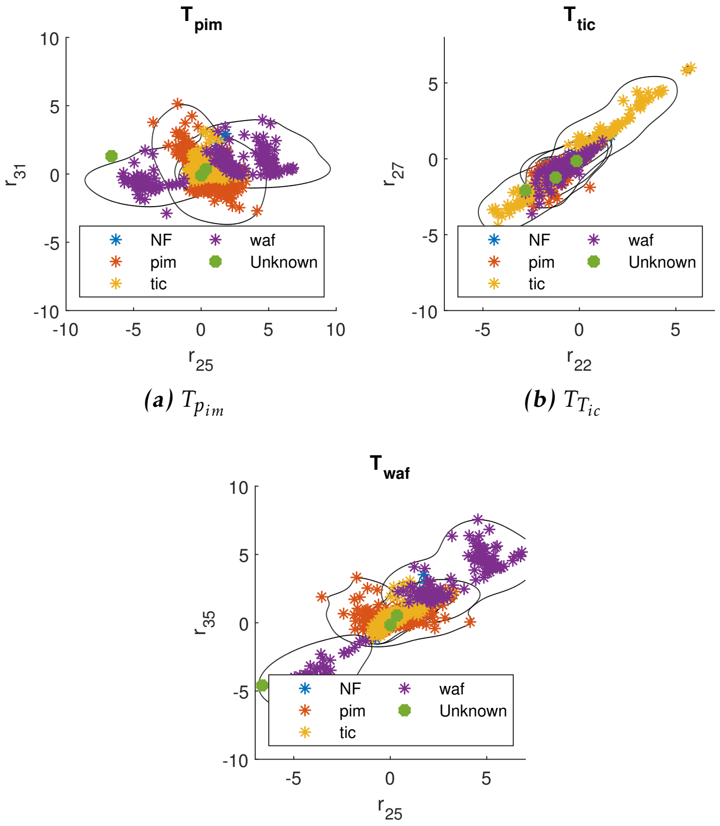
(d) $T_{W_{af}}$

Figure 4.8: *Support vector machine decision boundaries for all fault modes in classifiers* $\{T_{T_{ic}}, T_{p_{ic}}, T_{p_{im}}, T_{W_{af}}\}$*, using* $\nu = 0.5$ *and the radial basis function kernel.*

4.3 Data classification

The data classification was performed using the 1-class SVM in each classifier together with Eq.(3.8) as defined in Section 3.3. To illustrate the classification, samples from the three cases: NF, known fault mode and unknown fault mode are used. To test the unknown fault case, all data from $f_{p_{ic}}$ was removed from the training data and a single segment from $\{f_{p_{ic}}, -20\%\}$ was added. All in all, realizations of the following fault scenarios were added:

1. NF,
2. $f_{p_{ic}}, -20\%$ (Unknown),



***(a)** $T_{p_{im}}$*

***(b)** $T_{T_{ic}}$*

***(c)** $T_{W_{af}}$*

***Figure 4.9:** Distribution of new data in the $T_{T_{ic}}$, $T_{p_{im}}$ and $T_{W_{af}}$ classifier spaces.*

3. $f_{t_{ic}}$, +10% (known).

This test data is processed by the classifiers $\{T_{T_{ic}}, T_{p_{im}}, T_{W_{af}}\}$. $T_{p_{ic}}$ was removed since it is designed for $f_{p_{ic}}$. The distributions of the unknown data are displayed in Figure 4.9. The data from the unknown set $\{f_{p_{ic}} - 20\%\}$ is clearly separated from the nominal data in classifiers $\{T_{p_{im}}, T_{w_{af}}\}$. This is expected since looking at Table 4.2, these two classifiers utilizes the residual r_{25} which was also used for the $T_{p_{ic}}$ classifier.

The partial diagnoses along with the final diagnosis, as defined in Eq. (3.8), is shown for the different classifiers in Table 4.3. In this case, all of the test data fault modes were correctly isolated.

To examine the general novelty performance tests were conducted by removing one fault mode from the training data at a time, and using the remaining clas-

Table 4.3: *Partial diagnoses from classifiers $\{T_{T_{ic}}, T_{p_{im}}, T_{W_{af}}\}$, where the unknown data are samples from* NF, *$\{f_{p_{ic}}, -20\%\}$ and $\{f_{t_{ic}}, -10\%\}$. The final diagnoses are shown at the bottom of the table.*

Fault **Classifier**	NF	$f_{p_{ic}}, -20\%$	$f_{t_{ic}}, -10\%$
$T_{p_{im}}$	**NF**, f_{pim} f_{tic}, f_{waf}	$\emptyset$	NF, f_{pim} $\mathbf{f_{tic}}$, f_{waf}
$T_{t_{ic}}$	**NF**, f_{pim} f_{tic}, f_{waf}	NF, f_{pim} f_{tic}, f_{waf}	$\mathbf{f_{t_{ic}}}$
$T_{w_{af}}$	**NF**, f_{pim}, f_{tic}	$f_{w_{af}}$	NF, f_{pim}, $\mathbf{f_{tic}}$
Diagnosis	NF	UNKNOWN	$f_{t_{ic}}$

sifiers on the test data from the removed fault mode. The result is shown in Table 4.4, where the fraction of samples that were correctly classified as novelties are shown for each fault type and severity. The symbol "-" means that there were no data from that combination of fault and severity available.

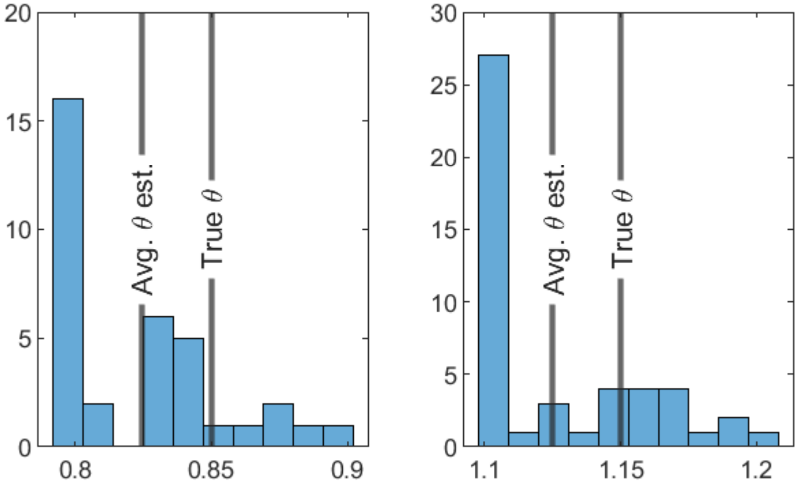
Table 4.4: *Novelty detection rate for the different fault severities when the corresponding fault mode was removed from training data.*

Severity **Fault mode**	-20	-15	-10	10	15	20
$f_{p_{ic}}$	0.70	-	-	0	-	-
$f_{W_{af}}$	1.00	-	0.25	-	-	0.95
$f_{T_{ic}}$	0.06	-	0.09	0.06	0.06	0
$f_{P_{im}}$	0.16	0.07	0.02	0.29	-	0.63

To assess the fault severity estimation, data was collected from the fault modes $\{f_{p_{im}}, -15\%\}$ and $\{f_{T_{ic}}, +15\%\}$. The data was gathered and evaluated according to the method described in Section 3.4. Each fault was collected from a full WLTC, and the results are presented in Table 4.5, using a segment length of 30 s and $k = 10$ in Eq. (3.12). In the table, detection rate is the fraction of validation samples where the true fault was a member of the final diagnosis $D_i \ni f_i$ and isolation rate is the fraction of validation samples where $D_i = f_i$. The average severity estimate is given in table but a closer examination of the distribution of the severity estimates are shown in Figure 4.10. The figure shows two histograms of the severity estimates along with the estimated $\hat{\theta}_i$ and the true θ_i. The reason why the leftmost bin is so tall in both cases is due to the severities represented in the training data. In the case of $\{f_{p_{im}}, -15\%\}$, the training data contains samples corresponding to $\theta_a = 0.8$ and $\theta_b = 0.9$ (note that the percentage error is given as $\theta - 1$). All elements in the leftmost bin corresponds to $\hat{\theta} = 0.8$, which means that all weights λ_i for distributions with severity θ_b is zero. The same line of reasoning holds for $\{f_{T_{ic}}, +15\%\}$ with $\theta_a = 1.1$ and $\theta_b = 1.2$.

***Table 4.5:** Table showing the performance of the classifier. Detection rate is the fraction of validation samples where the true fault is a member of the final diagnosis $D_i \ni f_i$ and isolation rate is the fraction of validation samples where $D_i = f_i$. This is presented along with the average severity estimate for the validation data.*

fault	severity	detection rate	isolation rate	avg severity est
$f_{p_{im}}$	−15 %	0.964	0.51	−17.1 %
$f_{T_{ic}}$	+15 %	0.909	0.84	+12.8 %



***(a)** $f_{p_{im}}$*

***(b)** $f_{T_{ic}}$*

***Figure 4.10:** Histogram of the estimated severities for fault modes $\{f_{p_{im}}$ − 15%$\}$ and $\{f_{T_{ic}}$, +15, %$\}$.*

5 Discussion

The goal of this chapter is to provide a discussion about the proposed method and the experimental results.

5.1 Result

The results from the initial testing suggest that Assumption 3.1 is consistent with the observed behaviour of the system. This is reflected in the severity estimates through the fact that, for all estimates it is observed that if $\theta_i \in [\theta_a, \theta_b]$ then $\hat{\theta}_i \in [\theta_a, \theta_b]$.

It should be noticed that the data used in algorithm testing is severely limited. In general, the amount of data available is of central importance for data-driven classifiers. When training data is limited it has a detrimental effect on classification performance and greatly increases the risk of overfitting. In this study, the training data totals roughly 20 h, which in the lifespan of a car (or even fleet of cars), is close to nothing. Thus, the results presented should not be considered a definite statement about the classifier performance, but rather a proof of concept that even when working with severely limited data, it is possible to apply the suggested algorithm with some success.

One major factor that influences the result is the behaviour of the residuals and thus the model they are generated from. Testing shows that the operating point has a significant influence on the model output error. This means that the operating point at which the data was collected induces a source of variance, separate from the present fault mode, which makes the comparison of data from different operating points more difficult. The result is that the classifier could end up being much better at identifying the operating point at which the data is collected, rather than the fault mode it is collected from. One way of addressing this issue would be to create a map of the model noise as a function of the operating point and use this in the normalization process. This could potentially eliminate the operating point induced variance. Two problems of this mapping is creating the actual map, which is likely time consuming, and the generalizability of the solution. Given that there are differences between engines, due to production or wear, the map from one engine might not be suitable for another engine even if they are of the same model. This problem also extends to the algorithm presented in this work. Both training data and validation data is collected from the same engine and thus further work is required to analyse the generalizability of the algorithm.

Another effect of the model variance is that it affects the stability of the residuals. In this case study, 24 out of the 35 generated residuals became unstable at higher loads. This means that residuals which are potentially useful at lower loads are discarded due to instability issues. This behaviour is closely tied to the engine model error. As stated earlier, if the model was perfect, the residuals would be perfect as well (disregarding measurement noise and numerical concerns). Since model development is a work intensive task, and thus an expensive one, model enhancement becomes a trade-off between cost and performance.

One part of the classification step is novelty detection. Table 4.4 clearly illustrates a result of selecting residuals to specifically detect only the faults which are present in the training data. Novelty detection rates vary significantly both between faults and between different severities of the same fault mode, but the detectability generally seems to increase with the magnitude of the fault severity ($|\theta - 1|$). One fault which stands out is $f_{T_{ic}}$, were the novelty detection is less than 10% for all severities. A reason for this is found in the residual selection. As shown in Table 4.2, the residual set $\{r_{17}, r_{22}\}$ is used to classify $f_{T_{ic}}$. Neither of these residuals are used in any of the other classifiers. This does not show that none of the residuals used in the other classifiers are sensitive to $f_{T_{ic}}$, but it at least shows that none of the residuals which are "best" at detecting $f_{T_{ic}}$ are used elsewhere.

5.2 Methodology

The 1-class SVM solutions used for classification limits the detection performance of known fault types with unknown severities. This is clearly illustrated in the $T_{p_{ic}}$ classifier as shown in Figure 4.7. The training data, of severity θ_1 and θ_2, is clearly separated which means that the fault mode is modelled as isolated islands. Testing suggests that any fault of this type with severity θ_i such that $\theta_1 < \theta_i < \theta_2$, would appear "between" these islands and if the overlap is insufficient, such faults are likely to be classified as unknown. This also holds for severities larger than that which is represented in training data, $\theta_i < \theta_1$ or $\theta_i > \theta_2$, as these would appear "beyond" the scope of the class. To address this issue, the severities used in training should be chosen to maximize the coverage in the classifier space. One alternate approach is to use multi-class SVM rather than 1-class SVM to model the fault modes. This partitions the entire residual space into regions where any data in a specific region is said to belong to a specific class. There are two main advantages to using 1-class SVM over this approach. The first is that it allows for overlapping classes potentially assigning data to more than one class. This is a desired property since there is considerable overlap of classes when it comes to the nominal case and fault modes which the classifier in question is not sensitive to. The second is that it allows for easy novelty detection for any data outside a modelled fault mode.

A problem that is related to fault mode modelling is the severity estimation. Given a λ^* which solves Eq (3.11) with a corresponding severity vector θ^*. The severity estimate is given as $\hat{\theta} = H(\lambda^*, \theta^*)$. The function $H(\cdot)$ used in this study is a weighted average, given by Eq (3.10). If more extensive fault modelling is used, this step could potentially be refined by choosing a more suitable transformation $H(\cdot)$.

Another property of the severity estimation is illustrated in Figure 4.10. As mentioned, the reason why the left bins are so tall is that they represent solutions where all non-zero elements of λ^* in Eq. (3.10) corresponds to distributions with the same severity. This is a direct result of using the KLD as cost function to minimize. Testing shows the the suggested method tends to be quite "radical" i.e. it often gives non-zero λ for a single severity, as illustrated in the figure. Given the proposed algorithm there might not be an obvious way to address this behaviour but one approach could be to take the size of the KLD into consideration. That is, assuming that data is unlikely to have the exact severity θ_i if it is "dissimilar" to all realisations of that severity, even though it might be less "dissimilar" to severity θ than another severity θ_j.

One goal of the residual selection process is to reduce the size of the input data while maintaining diagnostic performance. The method described in Section 3.1 does this by finding the residuals which is best at detecting the faults which are present in the training data. The novelty detection (described in Section 3.3) is used to detect anomalous data in the different classifier subspaces. The performance of this method is inherently tied to which residuals are actually used, or rather the sensitivity of these residuals. If none of the chosen residuals are sensitive to an unknown fault mode f_j, this fault will never be detected. Thus, by removing residuals, there is the risk of losing detectability.

6 Conclusions

This chapter summarizes some of the key conclusions that can be drawn from the study and discuss how these relate to the research questions as stipulated in Section 1.3. It also includes a brief discussion about some areas relating to the diagnostic framework that merits further study.

This work has outlined a diagnostic framework which combines a physical model with historical data to detect, classify and estimate the severity of, new faults in the system. The suggested method is not detailed in such a way that it can be readily implemented in an operating system. It should rather be considered as a proof of concept that the Kullback-Leibler divergence, together with historical data, can be used to estimate the severity of new faults in the system. It is also shown that a Gaussian parametric representation of data can be used to reduce the amount of storage required to maintain diagnostic performance.

Initial testing shows promising results but the data used for training was severely limited meaning that there is a significant need of further testing to evaluate the method. However, as stated in research question 1, the analysis of limited data was part of the purpose of the study. Considering the available data, it is not possible to come to any far reaching conclusions about classifier performance but the study shows some results that are worth noting. The two validation test shows that the fault was detected in 91 % and 96 % of the distributions respectively. This means that during the 28 min part of test cycle where the fault was induces, the classifier indicated that a fault was present for more than 25 min. The severity estimation shows that the estimated severity was in the correct span for both the tests but that the optimization tends to favour using few distributions for the estimation. As a result of this, estimates tend to be skewed towards one of the reference severities. This implies that there is room for improvement in this part of the algorithm.

Novelty detection was one of the goals of the study, as stated in research question 2. The proposed algorithm does not make any general claims about novelty detection though, and the novelty detection performance is closely tied to the residual selection. Since the method does not take the sensitivity of the residuals into account when the classifiers are constructed novelties will likely only be detected if any of the selected residuals happen to be sensitive to that specific fault.

Another goal of the study was to examine data representation, as specified in research question 3. The method applied was to segment the time series data and to estimate a Gaussian distribution for each segment. The conditions set for the experimental testing resulted in a compression rate in the order of 1000. From the perspective of data compression, this offers a significant reduction in the size of both training and validation data. Another advantage is that the parametrization facilitates the computation of the KLD. When it comes to performance, the method needs to be compared to other parametric models; something which is not covered in this study. One important consideration is that, as the dimensionality of the residual space increases, the performance might improve but it also becomes more difficult to estimate the distribution parameters.

The proposed method includes several design parameters, such as segment length, classifier dimensionality, kernel type and ν in the one 1-class SVM modelling. The interpretations and meanings of these parameters are discussed in this study but no strategy for parameter selection is given and is left for future study.

6.1 Future work

As shown by initial testing, the proposed method has potential as the basis of a diagnostic framework. There are however significant work needed before any such system could be used in practice. Some of the most interesting questions, which relate to the discussion in Section 5, are:

1. How does the system behave for other faults beside sensor faults? The experiments made in this study only includes multiplicative sensor faults and it would be interesting to examine how the framework behaves for other faults e.g. leakages, clogging or other type of sensor faults.

2. How can the method be extended to model the trend of data in the residual space? The proposed model tends towards using only one severity when the fault is estimated. It also does not cover any distributions "outside" the training data. A model that accounts for trends in the fault severity's impact on the distributions in the classifier spaces, could be examined to alleviate these limitations.

3. How should the fault severity be estimated, given a set $\{\lambda^*, \theta^*\}$? This is related to the previous question in the sense that a model of fault severity behaviour could improve upon the fault severity estimation performance.

Bibliography

[1] Michèle Basseville. Distance measures for signal processing and pattern recognition. *Signal Processing*, 18(4):349–369, dec 1989. ISSN 01651684. doi: 10.1016/0165-1684(89)90079-0. URL https://www.sciencedirect.com/science/article/pii/0165168489900790.

[2] Michèle Basseville. Divergence measures for statistical data processing - An annotated bibliography, apr 2013. ISSN 01651684. URL https://linkinghub.elsevier.com/retrieve/pii/S0165168412003222.

[3] Mariela Cerrada, René-Vinicio Sánchez, Chuan Li, Fannia Pacheco, Diego Cabrera, José Valente de Oliveira, and Rafael E. Vásquez. A review on data-driven fault severity assessment in rolling bearings. *Mechanical Systems and Signal Processing*, 99:169–196, jan 2018. ISSN 08883270. doi: 10.1016/j.ymssp.2017.06.012. URL https://linkinghub.elsevier.com/retrieve/pii/S0888327017303242.

[4] M. Dash and H. Liu. Feature selection for classification. *Intelligent Data Analysis*, 1(3):131–156, jan 1997. ISSN 1088467X. doi: 10.3233/IDA-1997-1302.

[5] Naiyang Deng, Yingjie Tian, and Chunhua Zhang. *Support vector machines: Optimization based theory, algorithms, and extensions*. CRC Press, jan 2012. ISBN 9781439857939. doi: 10.1201/b14297.

[6] John Duchi. Derivations for Linear Algebra and Optimization. pages 1–13, 1001. URL http://web.stanford.edu/{~}jduchi/projects/general{_}notes.pdf.

[7] J. L. Durrieu, J. Ph Thiran, and F. Kelly. Lower and upper bounds for approximation of the Kullback-Leibler divergence between Gaussian mixture models. In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, pages 4833–4836, 2012. ISBN 9781467300469. doi: 10.1109/ICASSP.2012.6289001.

[8] Larer Eriksson. Modeling and control of turbocharged SI and DI engines. In

Oil and Gas Science and Technology, volume 62, pages 523–538, 2007. doi: 10.2516/ogst:2007042.

[9] Lars Eriksson, Simon Frei, Christopher Onder, and Lino Guzzella. Control and optimization of turbocharged Spark ignited engines. In *IFAC Proceedings Volumes (IFAC-PapersOnline)*, volume 15, pages 283–288, 2002. doi: 10.3182/20020721-6-es-1901.01515.

[10] Erik Frisk, Mattias Krysander, Mattias Nyberg, and Jan Slund. A toolbox for design of diagnosis systems. In *Fault Detection, Supervision and Safety of Technical Processes 2006*, volume 1, pages 657–662. 2007. ISBN 9780080444857. doi: 10.1016/B978-008044485-7/50111-1.

[11] Andrea Giantomassi, Francesco Ferracuti, Sabrina Iarlori, Gianluca Ippoliti, and Sauro Longhi. Electric motor fault detection and diagnosis by kernel density estimation and kullback-leibler divergence based on stator current measurements. *IEEE Transactions on Industrial Electronics*, 62(3):1770–1780, mar 2015. ISSN 02780046. doi: 10.1109/TIE.2014.2370936.

[12] John Grezmak, Peng Wang, Chuang Sun, and Robert X. Gao. Explainable convolutional neural network for gearbox fault diagnosis. In *Procedia CIRP*, volume 80, pages 476–481, 2019. doi: 10.1016/j.procir.2018.12.008.

[13] Xiaojie Guo, Liang Chen, and Changqing Shen. Hierarchical adaptive deep convolution neural network and its application to bearing fault diagnosis. *Measurement: Journal of the International Measurement Confederation*, 93: 490–502, 2016. ISSN 02632241. doi: 10.1016/j.measurement.2016.07.054.

[14] Xiaojie Guo, Liang Chen, and Changqing Shen. Hierarchical adaptive deep convolution neural network and its application to bearing fault diagnosis. *Measurement: Journal of the International Measurement Confederation*, 93: 490–502, 2016. ISSN 02632241. doi: 10.1016/j.measurement.2016.07.054.

[15] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection, 2003. ISSN 15324435.

[16] Jinane Harmouche, Claude Delpha, and Demba Diallo. Incipient fault detection and diagnosis based on Kullback-Leibler divergence using principal component analysis: Part I. *Signal Processing*, 109:334–344, jan 2015. doi: 10.1016/j.sigpro.2014.06.023. URL `https://www.sciencedirect.com/science/article/pii/S0165168413002065`.

[17] Jinane Harmouche, Claude Delpha, and Demba Diallo. Incipient fault detection and diagnosis based on Kullback-Leibler divergence using principal component analysis: Part II. *Signal Processing*, 109:334–344, apr 2015. ISSN 01651684. doi: 10.1016/j.sigpro.2014.06.023. URL `https://www.sciencedirect.com/science/article/pii/S0165168414002898`.

[18] Jinane Harmouche, Claude Delpha, and Demba Diallo. Incipient fault amplitude estimation using KL divergence with a probabilistic approach. *Signal Processing*, 120:1–7, mar 2016. ISSN 01651684. doi: 10.1016/j.sigpro.

2015.08.008. URL https://linkinghub.elsevier.com/retrieve/pii/S0165168415002819.

[19] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *Springer Series in Statistics*, volume 27 of *Springer Series in Statistics*. Springer New York, New York, NY, 2009. ISBN 9780387848570. doi: 10.1007/b94608. URL http://www.springerlink.com/index/D7X7KX6772HQ2135.pdf.

[20] John R. Hershey and Peder A. Olsen. Approximating the Kullback Leibler divergence between Gaussian mixture models. In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, volume 4, 2007. ISBN 1424407281. doi: 10.1109/ICASSP.2007.366913.

[21] Andrew K.S. Jardine, Daming Lin, and Dragan Banjevic. A review on machinery diagnostics and prognostics implementing condition-based maintenance, 2006. ISSN 08883270.

[22] Feng Jia, Yaguo Lei, Jing Lin, Xin Zhou, and Na Lu. Deep neural networks: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data. *Mechanical Systems and Signal Processing*, 72-73:303–315, 2016. ISSN 10961216. doi: 10.1016/j.ymssp.2015.10.025.

[23] Daniel Jung and Erik Frisk. Residual selection for fault detection and isolation using convex optimization. *Automatica*, 97:143–149, nov 2018. ISSN 00051098. doi: 10.1016/j.automatica.2018.08.006. URL https://www.sciencedirect.com/science/article/pii/S0005109818303960?via{%}3Dihub.

[24] Daniel Jung and Christofer Sundstrom. A Combined Data-Driven and Model-Based Residual Selection Algorithm for Fault Detection and Isolation. *IEEE Transactions on Control Systems Technology*, 27(2):616–630, mar 2019. ISSN 1558-0865. doi: 10.1109/TCST.2017.2773514. URL https://ieeexplore.ieee.org/document/8122029/.

[25] C. T. Kelley. *Solving Nonlinear Equations with Newton's Method*. Society for Industrial and Applied Mathematics, 2003. ISBN 9780898715460. doi: 10.1137/1.9780898718898.

[26] S. Kullback and R. A. Leibler. On Information and Sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, mar 1951. ISSN 0003-4851. doi: 10.1214/aoms/1177729694.

[27] L Ljung. System identification. In *Signal Analysis and Prediction*, pages 163–173. 1998.

[28] P. Paris and F. Erdogan. A critical analysis of crack propagation laws. *Journal of Fluids Engineering, Transactions of the ASME*, 85(4):528–533, dec 1963. ISSN 1528901X. doi: 10.1115/1.3656900. URL http://fluidsengineering.asmedigitalcollection.asme.org/article.aspx?articleid=1431537.

[29] J Qian, T Hastie, J Friedman, Robert J. Tibshirani, and N Simon. Glmnet for Matlab, 2013. URL http://www.stanford.edu/{~}hastie/glmnet{_}matlab/.

[30] Chaitanya Sankavaram, Anuradha Kodali, Krishna R. Pattipati, and Satnam Singh. Incremental classifiers for data-driven fault diagnosis applied to automotive systems. *IEEE Access*, 3:407–419, 2015. ISSN 21693536. doi: 10.1109/ACCESS.2015.2422833. URL http://ieeexplore.ieee.org/document/7089165/.

[31] N. Sawalhi, R. B. Randall, and H. Endo. The enhancement of fault detection and diagnosis in rolling element bearings using minimum entropy deconvolution combined with spectral kurtosis. *Mechanical Systems and Signal Processing*, 21(6):2616–2633, 2007. ISSN 08883270. doi: 10.1016/j.ymssp.2006.12.002.

[32] Khalid. Sayood. *Introduction to Data Compression: .* Morgan Kaufmann Publishers Inc, dec 2005. ISBN 9780080509259. URL https://www.dawsonera.com:443/abstract/9780080509259.

[33] Bernhard Schölkopf, Robert Williamson, Alex Smola, John Shawe-Taylor, and John Piatt. Support vector method for novelty detection. In *Advances in Neural Information Processing Systems*, pages 582–588, 2000. ISBN 0262194503.

[34] Bernhard Schölkopf, John C. Platt, John Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001. ISSN 08997667. doi: 10.1162/089976601750264965. URL http://web.a.ebscohost.com/ehost/pdfviewer/pdfviewer?vid=1{&}sid=925977c1-0308-464a-9930-f2f741391dca{%}40sessionmgr4007.

[35] Ingo Steinwart and Andreas Christmann. *Support vector machines*. Information science and statistics. Springer, 1st ed. edition, 2008. ISBN 9780387772424.

[36] Monica Tutuianu, Alessandro Marotta, Heinz Steven, Eva Ericsson, Takahiro Haniu, Noriyuki Ichikawa, and Hajime Ishii. Development of a World-wide Worldwide harmonized Light duty driving Test Cycle. *Technical Report*, 03 (January):7–10, 2014.

[37] Venkat Venkatasubramanian, Raghunathan Rengaswamy, and Surya N. Kavuri. A review of process fault detection and diagnosis part II: Qualitative models and search strategies. *Computers and Chemical Engineering*, 27(3): 313–326, 2003. ISSN 00981354. doi: 10.1016/S0098-1354(02)00161-8.

[38] Venkat Venkatasubramanian, Raghunathan Rengaswamy, Surya N. Kavuri, and Kewen Yin. A review of process fault detection and diagnosis part III: Process history based methods. *Computers and Chemical Engineering*, 27 (3):327–346, 2003. ISSN 00981354. doi: 10.1016/S0098-1354(02)00162-X.

[39] Venkat Venkatasubramanian, Raghunathan Rengaswamy, Kewen Yin, and Surya N. Kavuri. A review of process fault detection and diagnosis part I: Quantitative model-based methods. *Computers and Chemical Engineering*, 27(3):293–311, 2003. ISSN 00981354. doi: 10.1016/S0098-1354(02)00160-6.

[40] E. M. Wright and Richard Bellman. Adaptive Control Processes: A Guided Tour. *The Mathematical Gazette*, 46(356):160, may 1962. ISSN 00255572. doi: 10.2307/3611672. URL `https://www.jstor.org/stable/3611672?origin=crossref`.

[41] Abdulrahman Youssef, Claude Delpha, and Demba Diallo. An optimal fault detection threshold for early detection using Kullback-Leibler Divergence for unknown distribution data. *Signal Processing*, 120:266–279, mar 2016. ISSN 01651684. doi: 10.1016/j.sigpro.2015.09.008. URL `https://linkinghub.elsevier.com/retrieve/pii/S0165168415003102`.

[42] Jiusun Zeng, Uwe Kruger, Jaap Geluk, Xun Wang, and Lei Xie. Detecting abnormal situations using the Kullback-Leibler divergence. *Automatica*, 50(11):2777–2786, nov 2014. ISSN 00051098. doi: 10.1016/j.automatica.2014.09.005. URL `https://linkinghub.elsevier.com/retrieve/pii/S0005109814003598`.

[43] Fan Zhang, Yu Liu, Chujie Chen, Yan Feng Li, and Hong Zhong Huang. Fault diagnosis of rotating machinery based on kernel density estimation and Kullback-Leibler divergence. *Journal of Mechanical Science and Technology*, 28(11):4441–4454, nov 2014. ISSN 1738494X. doi: 10.1007/s12206-014-1012-7.

[44] D. Zogg, E. Shafai, and H. P. Geering. Fault diagnosis for heat pumps with parameter identification and clustering. *Control Engineering Practice*, 14(12):1435–1444, 2006. ISSN 09670661. doi: 10.1016/j.conengprac.2005.11.002.