Optimisation du système de surveillance des hélicoptères pour l'amélioration du diagnostic et de la maintenance

Johan Wiig

N°: 2006 ENAM 55

Ecole doctorale n° 432 : Sciences des Métiers de l'Ingénieur

T H È S E

pour obtenir le grade de

Docteur

de

l'École Nationale Supérieure d'Arts et Métiers

Spécialité "Automatique"

présentée et soutenue publiquement
par

Johan WIIG

le 11 décembre 2006

OPTIMIZATION OF FAULT DIAGNOSIS IN HELICOPTER HEALTH AND USAGE MONITORING SYSTEMS

Directeur de thèse : ***Daniel BRUN-PICARD***

Codirecteur de thèse : ***Hassan NOURA***

Jury :

Sylviane GENTIL, Professeur, INPG / ENSIEG, Grenoble ..... Rapporteur
Dominique SAUTER, Professeur, Université Henri Poincaré, Nancy I ..... Rapporteur

Pierre-Antoine AUBOURG, Ingénieur, Chef de service, Eurocopter ..... Invité
Daniel BRUN-PICARD, Professeur, ENSAM, Aix en Provence ..... Examinateur
Mathieu GLADE, Docteur, Chef d'équipe, Eurocopter ..... Examinateur
Hassan NOURA, Professeur, Université Paul Cézanne, Aix-Marseille III ..... Examinateur
Mustapha OULADSINE, Professeur, Université Paul Cézanne, Aix-Marseille III .. Examinateur
Michel VERGÉ, Professeur, ENSAM, Paris ..... Examinateur

Laboratoire des Sciences de l'Information et des Systèmes
LSIS – UMR CNRS 6168

Acknowledgments

This study has been realized at Laboratoire des Sciences de l'Information et des Systèmes (UMR CNRS 6168) and Ecole Nationale Supérieure des Arts et Métiers. The subject and industrial context was provided by Eurocopter, with funding through the Marie Curie Host Fellowship.

First and foremost I would like to thank my advisors Daniel BRUN-PICARD, professor at LSIS, ENSAM, Aix en Provence, and Hassan NOURA, professor at LSIS, Université Paul Cézanne, Marseille.

Further, I would like to thank :

Sylviane GENTIL, professor at INPG, ENSIEG, Grenoble, and Domunique SAUTER, professor at Université Henri Poincaré, Nancy, for accepting to participate as reporters on the jury, as well as for their remarks and suggestions to my manuscript.

Michel VERGÉ, professor at ENSAM, Paris, for accepting to participate as examinator on the jury, as well as for his remarks and suggestions to my manuscript.

Pierre-Antoine AUBOURG at Eurocopter for accepting to participate on the jury, and for validating the industrial aspects of my work.

Mathieu GLADE at Eurocopter for accepting to participate as examinator on the jury, and for supporting me through the final year of my study.

Mustapha OULADSINE, professor at LSIS, Université Paul Cézanne, Marseille, for accepting to participate as examinator on the jury.

Luc DAURES, Tran BANG, Philippe JOLY, Cecile ALEXANDRE, Jean-Charles ANIFRANI and Jean-Pierre DERAIN at Eurocopter for their help and support.

Finally, I would like to thank my family and friends, as well as everyone else concerned at LSIS, ENSAM and Eurocopter, for their help and support in the completion of this study.

Contents

Acronyms

ARMS Aircraft Recording and Monitoring System

CAA Civil Aviation Authority (UK)

CBM Condition Based Maintenance

CG Center of Gravity

COTS Commercial Off The Shelf

CVR Cockpit Voice Reorder

DFT Discrete Fourier Transform

EuroARMS Eurocopter Aircraft Recording and Monitoring System

EVM Engine Vibration Monitoring

FAA Federal Aviation Authority (USA)

FDR Flight Data Recorder

HARP Helicopter Airworthiness Requirements Panel

HUMS Health and Usage Monitoring System

IAS Indicated Airspeed

IGB Intermediate Gear Box

IT Information Technology

KTS Knots

LPC Linear Predictive Coding

LCC Life Cycle Cost

MARMS Modular Aircraft Recording and Monitoring System

MGB Main Gear Box

MMH/FH Mean Man Hours / Flight Hours (maintenance)

MTBF Mean Time Between Failure

NF Engine turbine speed

NG Engine generator speed

OEM Original Equipment Manufacturer

PAC Power Assurance Check

RTB Rotor Track and Balance

SHL Steward Hughes Limited

TBM Time Based Maintenance

TBO Time Between Overhaul

TDS Tail Drive Shaft

TGB Tail Gear Box

VMS Vibration Monitoring System

VPN Virtual Private Network

Chapter 1

Introduction

Increasing demand for both reduced rotorcraft maintenance cost and improved operational safety has paved the way for the Health and usage Monitoring System (HUMS). These systems emerged in the early nineties as a response to the relatively high accident rate experienced by offshore shuttle helicopters trafficking the petrol installations in the North Sea. However, it soon became clear that these systems, in addition to reducing accident rates, had a potential for maintenance cost reduction. Research and development into HUMS technologies over the years has kept a focus on both aspects by working toward better safety. At the same time, efforts have been made to exploit the increased situation awareness given by the HUMS in order to help the operators better organize their maintenance tasks.

A HUMS deploys both proactive and reactive methods to anticipate drivetrain failure. Proactive methods include usage spectrum analysis such as load cycle calculation, allowing remaining component safe life to be estimated based on the actual stress a component has been under for the duration of its service. The reactive approach is based on detecting propagating component failure at an early stage, before seizure occurs. This method relies on a sensor network covering engines and transmission system. For the current generation HUMS, this sensor network is mainly limited to vibration sensors and angular shaft speed sensors.

During operation, the HUMS airborne segment gathers data from its sensor network. Some HUMS performs diagnosis real time in flight, providing the pilots with instant warning of any suspected problems. However, most HUMS perform diagnosis and reporting between flights. This is achieved by transferring the data, by means of a data cartridge, to a stationary computer. The stationary computer, known as a ground station, analyzes the recorded data and produces a discrepancy report for the maintenance crew.

This study is concerned with methods to interpret the vibratory data as

accurately as possible. The motivation for this is twofold; increased safety and reduced maintenance cost. By improving the detection capabilities of the system, the risk of in-flight mechanical failure is reduced. As a rotorcraft drive-train is largely non-redundant, failure can have serious consequences. Further, all HUMS, like any automated fault detection system, are prone to produce unjustified alerts from time to time. This has implications both on the operational availability as well as on the maintenance cost of the rotorcraft, as false alarms often results in unnecessary aircraft grounding and maintenance. The methods described in this study are designed to produce vibration-base diagnosis accurately as possible, so that the fault detection rate is maximized and the false alarm rate minimized. An additional objective is removing any aircraft specific configuration of the HUMS. The need for configuring, or training, the HUMS for each aircraft, and retrain it after major overhauls, is a weak-spot on most commercial HUMS. This imposes a significant workload on the operator, and renders the HUMS vulnerable miss-training. Both of which detract from the system's usefulness by increased operating cost and reduced fault detection capabilities.

This report is organized in 8 chapters. Chapter 1 contains the general introduction to the subject. More detail on HUMS is given in chapter 2, with chapter 3 detailing the state of the art for the technologies deployed in a HUMS. Practical issues concerning data transfer and storage are elaborated in chapter 4. Chapter 5 treats validation and pre-processing of HUMS vibration data. New methods for feature extraction are developed in chapter 6, and fault detection in chapter 7. Finally, concluding remarks are presented in chapter 8. In order to keep the report as brief and clear as possible, details on mathematical tools are kept in the appendixes.

Chapter 2

Problem Statement

2.1 Background

2.1.1 History

The history of Health and Usage Monitoring System (HUMS) dates back as far as the mid eighties. At this time, it became clear that helicopters operated in the North Sea where overrepresented in the accident statistics, compared to equal size turbo prop airplanes. The UK Civil Aviation Authority (UK) (CAA) Helicopter Airworthiness Requirements Panel (HARP) submitted a report in 1986, concluding that the risk level in North Sea helicopter operations were above what could be seen as acceptable [47]. To improve rotorcraft airworthiness, several steps were recommended. Among them was the permanent installation of vibration monitoring equipment.

Vibration monitoring of mechanical systems was at the time already an established technology. Although not previously tested on aircraft, such techniques had already proven their effectiveness in condition monitoring of industrial machinery, such as paper mills and power plants. However, it was not until the eighties that the size and weight of the necessary numeric hardware were in such a manner that it could be fitted on a helicopter.

By the end of the eighties, two parallel trials were under way. Motivated by the HARP report, and largely sponsored by the petroleum industry, these programs aimed at testing the concept of in-flight vibration monitoring. One of the programs was led by Steward Hughes Limited (SHL) / Teledyne, the other by Meggitt Avionics. The purpose of the trials was however more to create a proof of concept, than refining diagnosis algorithms. By the time the trials ended in 1991, the technology was, however promising, still regarded as immature.

In 1990, CAA issued new regulations making Flight Data Recorders

(FDRs) mandatory in helicopters operating in hostile environments. The avionics manufacturers participating in the HUMS trials saw this as an opportunity to introduce their newly developed technology to the market. With the operators' and the petroleum industry's increasing interest in the technology, creating combined FDR / Cockpit Voice Reorder (CVR) / HUMS systems had obvious competitive advantages. Thus, two FDR / CVR / HUMS systems were put on the market; SHL's North Sea HUMS and Meggitt Avionics' IHUMS.

Al though not mandatory by law, the oil companies' great interest in these systems made them an important burgeoning point when negotiating service contracts with the rotorcraft operators. As a result, HUMS quickly became a reality for all operators involved in offshore flight, on both sides of the North Sea. In 1999, the CAA issued regulations making HUMS mandatory for all heavy rotorcraft registered in the UK.

2.1.2 Motivations

As already mentioned, the initial motivation for introducing vibration monitoring in helicopters was safety. However, it soon became clear that a tool capable of describing the actual condition of critical components had considerable potential in maintenance planning and cost reduction.

Aircraft maintenance workload is normally measured in Mean Man Hours / Flight Hours (maintenance) (MMH/FH). Maintenance workload is highly dependent on aircraft size and type, and can be found anywhere from less than one hour to several hours per flight hour. Compared to equal size turbo prop airplanes, even the most economical rotorcrafts have very high operating cost due to maintenance. In fact, around 25% of the total life cycle cost (LCC) for most helicopters is maintenance, equivalent to the acquisition cost.

Maintenance can be divided in two main categories: Condition Based Maintenance (CBM), and Time Based Maintenance (TBM). CBM represents the maintenance tasks which are generated as a result of faults uncovered during inspections, faults uncovered by HUMS, and operational irregularities, such as torque limit overshoots or rotor over-speeds. TBM, on the other hand, is performed at various fixed intervals. Some are just a few hours apart, or even between each flight. This is the tedious day-to-day work of inspections, to ensure that the aircraft is in an airworthy condition.

The TBM workload is very high on rotorcraft compared to most other vehicles, and is one of the main cost drivers in helicopter operations. This is due to the large amount of moving parts in the helicopter transmission system, as well as the lack of redundancy in the power path from engine

to rotor. Because of the lack of redundancy, several failure modes in the helicopter transmission system can be catastrophic. To minimize risk, very strict and expensive maintenance routines must be followed in rotorcraft TBM.

Every component on a helicopter has a safe life limit. Upon reaching this age, the component must be overhauled. The safe life limit of each component is derived from an expected usage spectrum of the aircraft, and then given a substantial margin. Consequently, most retired parts are in a perfectly good condition. However, if an aircraft is exposed to higher loads than what was anticipated when the maintenance schedules where created, components might be exposed to more stress than they where design to handle (Fig. 2.1).

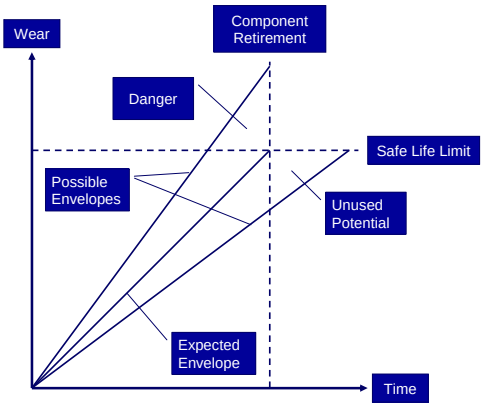

Figure 2.1: HUMS Overview

Most of the inspections and overhauls performed as TBM are unnecessary, in the sense that maintenance is performed on helicopters which are in a perfectly airworthy condition. This is however the proactive nature of TBM, if one is to ensure that the possibility of mechanical failure is minimized. Obviously, helicopter operating costs could be decreased dramatically if one were to perform maintenance only "on condition" (CBM), whenever a failure occurs. However, performing corrective maintenance after a fault has occurred will in most cases pose an unacceptable safety risk.

This is, of course, unless one has a reliable way, other than manual inspection, to detect a propagating fault before it becomes critical. HUMS was, and still is, regarded as the answer to this problem. In addition to increase safety, HUMS was seen as the technology that would revolutionize rotorcraft maintenance, and shift rotorcraft maintenance strategy from TBM to CBM. For various reasons, these ambitions have so far not been reached.

2.1.3 Regulatory Definition

The only formal definition of HUMS is maintained by the UK CAA, as the UK is still the only country where HUMS is mandatory. HUMS is mandatory for helicopters in the following category:

> United Kingdom registered helicopters issued with a Certificate of Airworthiness in the Transport Category (Passenger), which have a maximum approved seating configuration of more than 9 passengers.

In reality, HUMS is demanded on all offshore flights by the petroleum companies operating in the North Sea. Consequently, HUMS becomes a requirement for heavy helicopters operating in both British and Norwegian sector.

The CAA definition divides HUMS in two main subsystems; a Vibration Monitoring System (VMS), and "existing established techniques". The latter part covers functions such as temperature- and torque monitoring, magnetic plugs and chip detectors, thus corresponding to the Usage Monitoring System (UMS) of HUMS. It is worth noting that these functions are mandatory on all helicopters, regardless of whether a HUMS is installed or not. In case no HUMS is installed, these functions are maintained by other systems.

The VMS addresses the Health aspect of HUMS. The definition applies to all rotorcraft, and is thus not very precise. The CAA directive [1] reads as follows

> Vibration monitoring System (VMS) should monitor :
>
> - Engine to main gearbox input drive shafts
> - Main gearbox shafts, gears and bearings
> - Accessory gears, shafts and bearings
> - Tail rotor drive shafts and hanger bearings
> - Intermediate and tail gearbox gears, shafts and bearings
> - Oil cooler drive
> - Main and tail rotor track and balance

Further, the HUMS Minimum Equipment List (MEL) states that

> Depending upon system installation, if the data analysis (or failure indication system) indicate a malfunction of any system or

sensor, i.e. accelerometer, then the maximum period that the item or system can be deemed unserviceable would be as follows:

(1) 25 flying hours

However, if the specific item has been under investigation due to adverse trend identified by the HUM system, the maximum period of unserviceability would be as follows:

(2) 10 flying hours

2.2 Rotorcraft Failure Modes

The transmission system of a heavy rotorcraft is highly complex, and has a high number of possible failure modes. Failure scenarios are typically complex, in the sense that one propagating fault tends to trigger other failures. This is especially the case for gearboxes. Still, it is possible to distinguish some classical fault types, and their symptoms.

2.2.1 Engines

Helicopter jet engines consist of two stages. The first stage includes compressor, combustion chamber and turbine, and resembles the design of a traditional fixed-wing engine. This assembly is followed by the second stage, which is an additional turbine. The second stage delivers power from the engine to the transmission system.

Engine Compressor and Turbine Unbalance

Engine turbines and compressors rotate at very high speeds, and must be perfectly balanced. Problems like disk cracks, blade cracks and broken blades typically produce unbalanced rotation, and are uncovered by monitoring vibration energy at the frequencies corresponding to the compressor and turbine rotating speeds.

Engine Power Degradation

Engine performance is gradually degraded throughout the lifetime of the engine. Performance must however not be allowed to drop below a certain minimum threshold. The potential of an engine is determined by measuring which engine temperature is required to sustain a given torque.

2.2.2 Transmission System

The helicopter transmission system is a set of shafts and gearboxes which receives the power from the engines, and forwards it to the rotors as well as equipment such as cooling fans, hydraulic pumps and power generators. Helicopter transmission systems are characterized by high exchange rates, high power and low weight, making it critical high-precision machinery.

Shaft Unbalance

Unbalanced rotation is caused by bent or otherwise damaged shafts. This fault types is especially critical for the high speed shafts between the engines and the main gearbox, due to the amount of force generated by even slight unbalance. Shaft unbalance is easily detectable by measuring the energy corresponding to the shaft rotating speed.

Shaft Misalignment

Bad shaft coupling can cause one shaft element to become misaligned. This is a critical point for engine shafts, and for rotorcraft where the tail drive shaft is made up of concatenated segments. Shaft unbalance is easily detectable by measuring the energy corresponding to twice the shaft rotating speed.

Localized Gear Damage

Bent or broken gear teeth are critical faults in helicopter transmission systems, and require immediate retirement of the damaged components. A localized damage to a gear surface typically generates an irregularity in the vibration waveform whenever the damaged region interfaces with another gear.

Gear Hub Crack

A gear hub / web crack occurs, like localized damage, due to excessive load. This fault type will change the gear shape from perfect circular to somewhat oval. An oval rotating track results in the vibration signature changing periodically, with period equal to the gear rotation.

Distributed Gear Damage

Distributed gear damage, or fretting, occurs as fine scratches across the gear tooth pattern. This fault type typically accompanies localized damage and hub cracks, as unbalanced rotation or rough tooth edges on one gear tend

to damage its interfacing gear. Distributed gear damage alters the vibration signature just slightly, and is inherently difficult to detect using traditional vibration monitoring.

Epicyclic Carrier Cracks

Several rotorcraft gearboxes have one or two epicyclic stages just before the output to the main rotor. Due to high torque, the epicyclic planet carrier is in some cases prone to crack propagation. Cracks from the carrier rim toward the center results in one or more planets interfacing with different force than the others. This will again result in a periodic change in vibration signature, with period equal to the carrier rotation.

Bearing Race and Roller Cracks

Excessive load to bearings might cause cracks in the races or the rolling elements themselves. A crack in a bearing race will generate a sharp pulse whenever a roller passes over it. A roller crack will generate a pulse whenever the roller interfaces with the inner or outer race.

Bearing Corrosion

Corrosion is a problem for external bearing, such as those on the tail drive shaft. Corroded bearing races typically produce wide band noise, creating an increase in the total vibration energy.

2.2.3 Rotors

The rotors are the non-redundant lifting and anti-torque devices of a helicopter. Control and propulsion is also managed by the rotors, making them the most critical part of any helicopter. Serious rotor damage is usually catastrophic.

Unbalance

A rotor must have its Center of Gravity (CG) at the center of the mast to avoid vibration at the frequency corresponding to the mast rotating speed. The amplitude of an unbalance is identified by measuring vibration in the rotor plane. Unbalance direction (which blades are too light or too heavy) can be measured using a blade indexer. Correction is done by simply adding or removing weights from the blades.

Blade Track Split

All the blades of a rotor should ideally follow the same track. Worn blades do however tend to change track slightly, causing increased vibration levels. Blade track can be estimated either by measuring vertical acceleration, or by using a camera measuring the distance between each blade and the airframe. Correction is done by altering length of the blade pitch links, or by using bendable flaps on the blades.

Bearing Wear

Some rotorcraft use fully articulate rotors, meaning that each blade can be rotated along three axis. This is achieved by fixing the blade sleeve to the hub using an assembly of three traditional bearings, or a single spheric elastomer bearing. The former solution is prone to traditional bearing problems, while the latter might suffer from damaged elastomer. Such problems are identifiable by an increase in vibration energy at the frequency corresponding to the mast rotating speed or blade pass speed.

Damper Wear

Fully articulate rotors use dampers on the main rotor to damp blade movement in the lead / lag (horizontal) plane. Worn dampers are identifiable by an increase in vibration energy at the frequency corresponding to the mast rotating speed or blade pass speed.

Swashplate Eccentricity

The swashplate is basically a gigantic bearing which encircles the main rotor mast. One part of the swashplate is fixed to a set of actuators mounted on the top of the main gearbox. The other part is connected to the blade pitch links. The swashplate is the medium allowing the stationary actuators to control the angle of the rotating blades.

Like all bearings, the swashplate is prone to generate some eccentricity after excessive use. This is usually detectable as an increase in vibration energy at the frequency corresponding to the mast rotating speed or blade pass speed.

2.3 Health and Usage Monitoring Tasks

A HUMS is responsible for alerting the operator of any problems which might threaten the airworthiness of the aircraft. To accomplish this, the HUMS

use a sensor array covering most critical components on the aircraft. Some of these sensors are part of the standard avionics package, such as airspeed sensors and engine temperature probes. Others, like accelerometers and rotor indexers, are proprietary to the HUMS.

A HUMS works both reactively and proactively. The reactive approach allows the HUMS to detect any faults present in rotorcraft, while the proactive methods allow faults to be anticipated before they occur.

2.3.1 Sensors and Acquisition Procedures

The vibration monitoring part of a HUMS uses three types of data; accelerometer and tachometer signals, as well as contextual parameters such as airspeed, temperature and torque. The need for the latter category of data will be explained later. Accelerometers are mounted on all critical components, including gearboxes, engines and the bearing block for the tail drive shaft (Fig. 2.2). The rotors are covered by accelerometers mounted on the airframe. Speed sensors are mounted on each engine compressor, engine output turbine, and on each rotor. The rotor speed sensors generate only one pulse per rotation, making it possible to know the position of the rotors relative to the vibration phase.

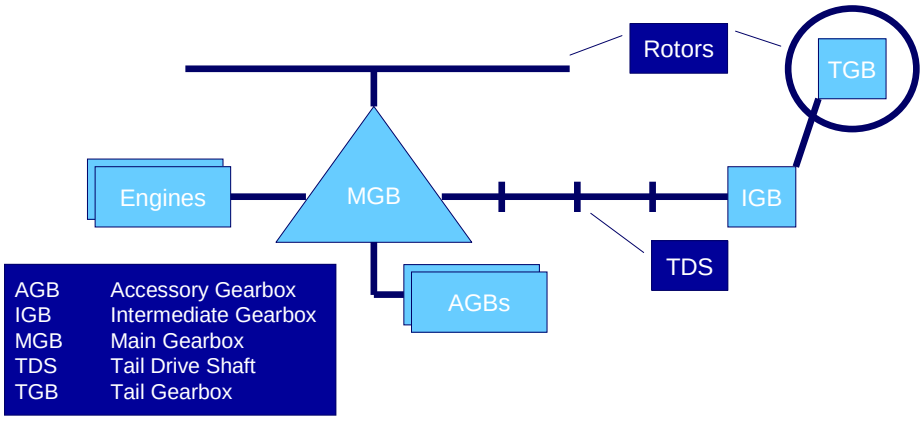

Figure 2.2: Mechanical Overview

A HUMS solution for a large aircraft can require more than thirty accelerometers, making it impossible to acquire all accelerometers simultaneously without generating enormous volumes of data. To counter this, all commercial HUMS solutions acquire only one component at a time with a finite length acquisition. During flight, the HUMS airborne segment cycles a preset program acquiring data from all components, one at a time.

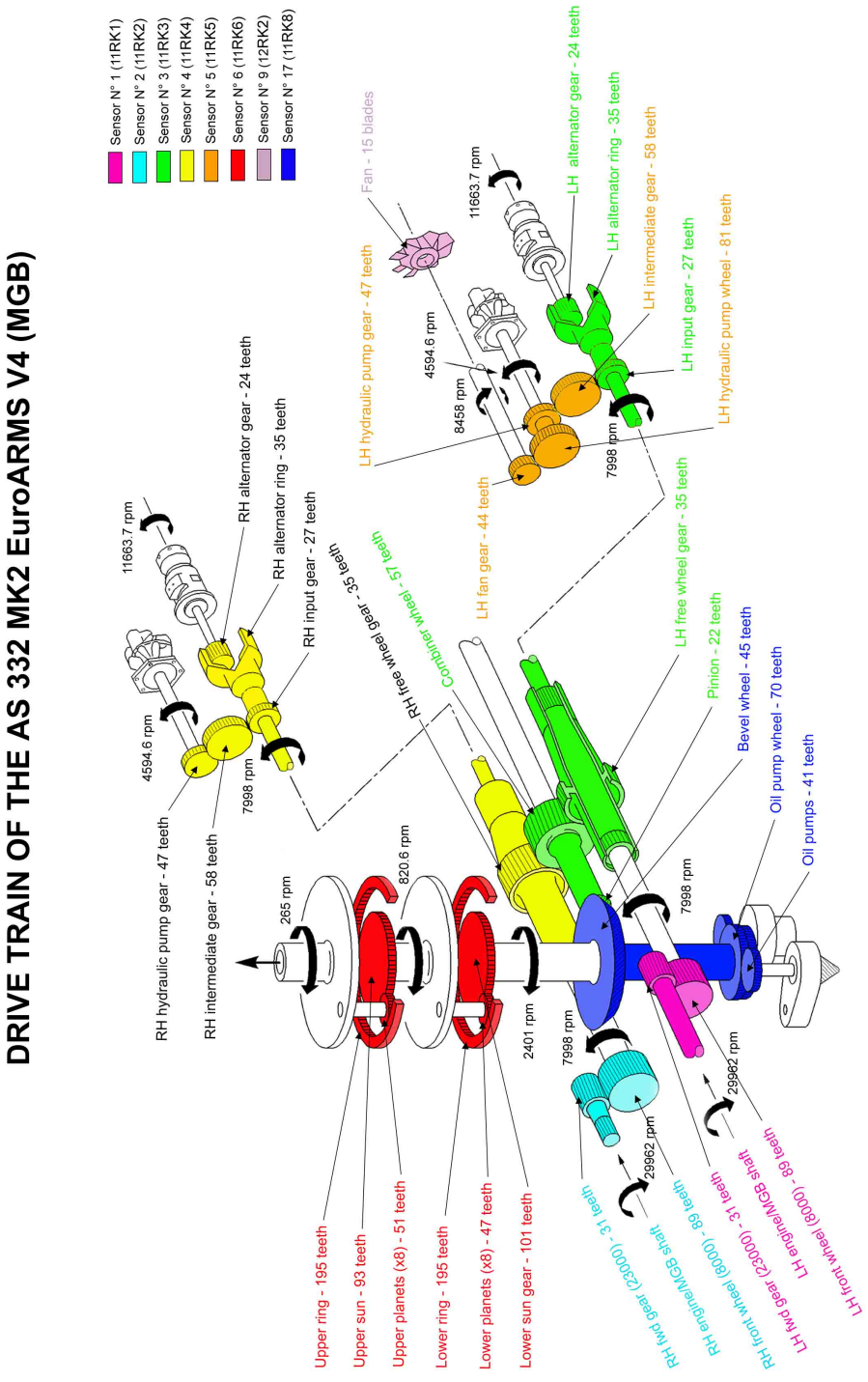

Figure 2.3: Mechanical overview for AS332L2, part 1.

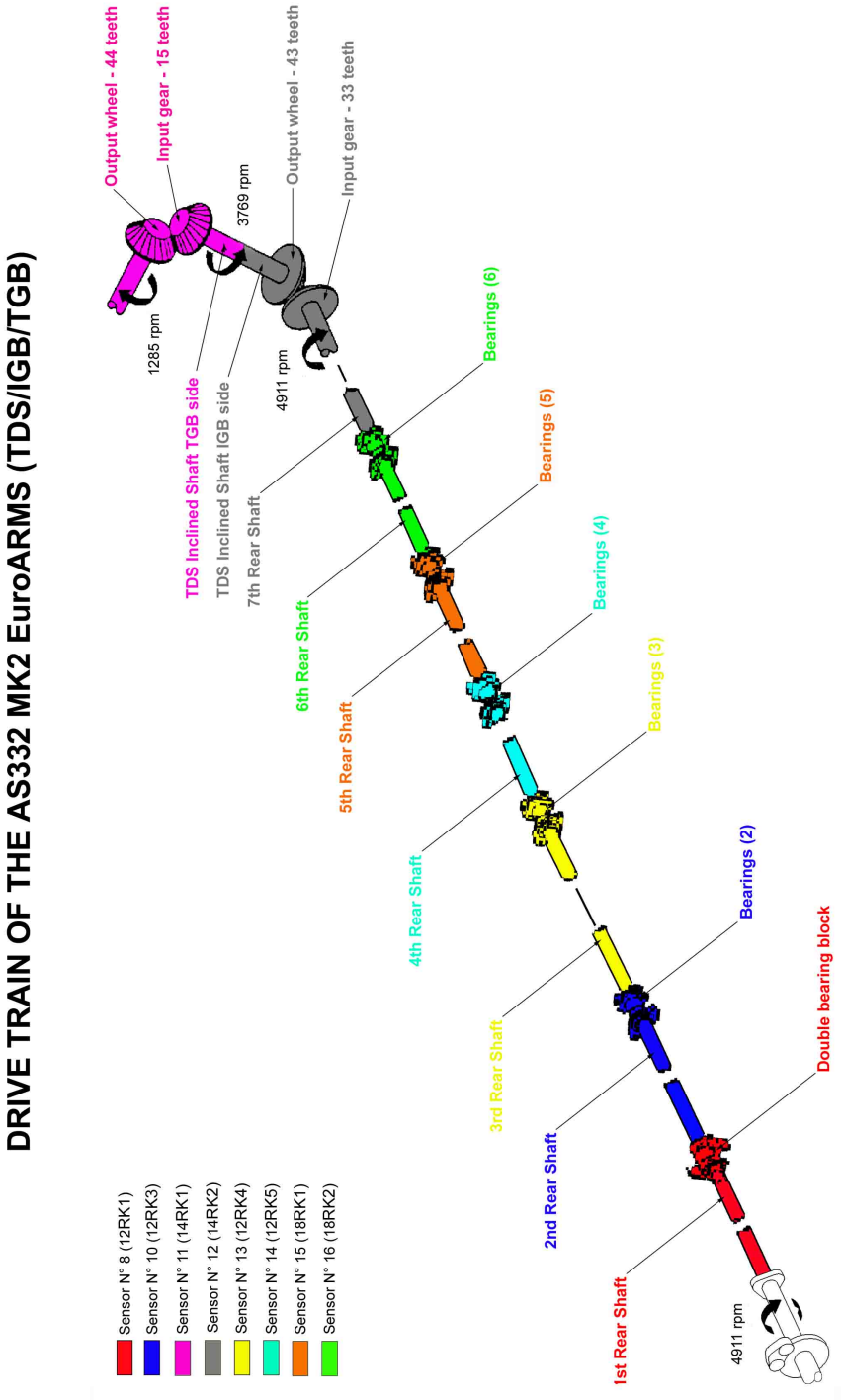

Figure 2.4: Mechanical overview for AS332L2, part 2.

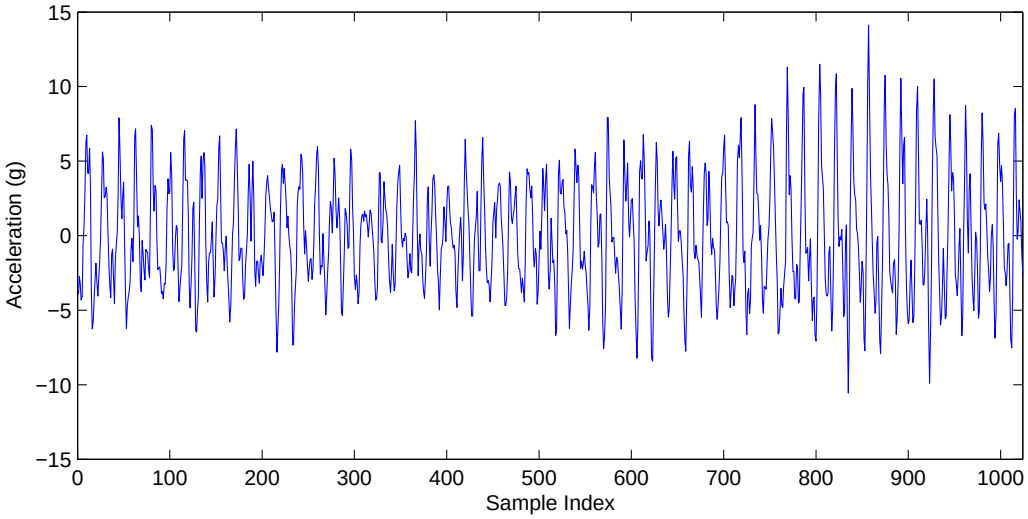

Figure 2.5: AS332L2 left hand ancillary intermediate gear acquisition.

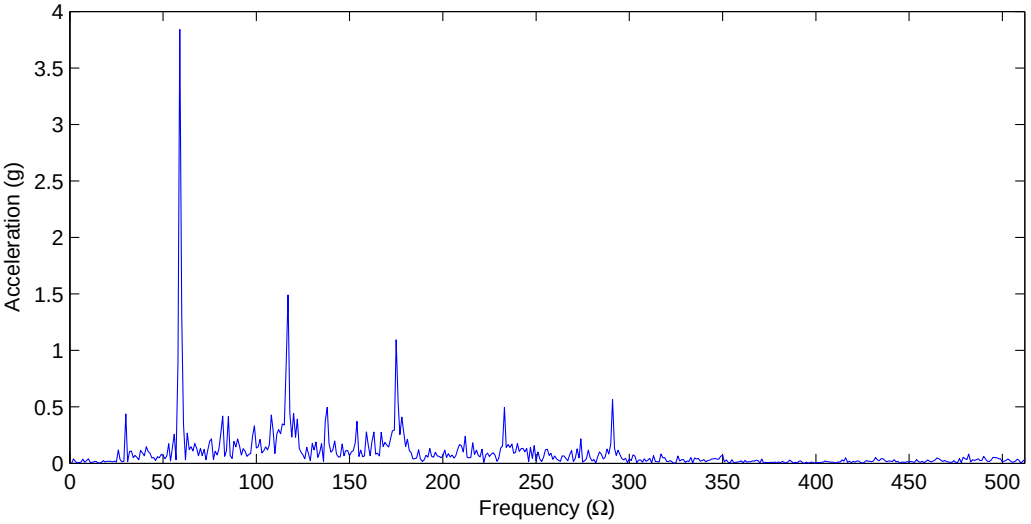

Figure 2.6: AS332L2 left hand ancillary intermediate gear acquisition.

As the transmission system of every rotorcraft is different, so is the sensor positioning. Figures 2.3 and 2.4 shows sensor positioning for the HUMS EuroARMS when fitted on a AS332L2 rotorcraft. Figures 2.5 and 2.6 shows an acquisition from the AS332L2 left hand ancillary intermediate gear in the temporal and the frequency domain.

2.3.2 Usage Monitoring

Two proactive methods exist. One is estimating the load on key each components, and integrating this over time to see the total stress the components have been subjected to. This allows the HUMS to estimate the remaining safe life limit for the components. The other method is simply detecting obvious misuse, such as engine overloads and over speeds.

Parameter Exceedance

Which parameters to monitor for exceedances and how to monitor them is aircraft dependent, but usually involves engine torque, engine temperature and rotor speed. Parameter threshold overshoots are automatically logged by the HUMS, together with additional information such as time, time over threshold, max value, etc. Traditional aircraft avionics displays a warning directly to the pilots whenever an event is detected. The pilots must then relay this information to the maintenance crew. The advantage of HUMS when recording excessive use is automated logging, more precise logging, as well as logging of additional information which helps determine the seriousness of the event, and consequently the best choice of corrective maintenance.

Load Cycle Calculation

All rotorcraft parts have a safe life limit. For most parts, the safe life limit is defined in flight time. For more critical components, mainly engines, rotors and gearboxes, a safe life limit is also defined in load cycles. Al though flight time gives a good pointer to the total stress a component has been subject to, flight time does not reflect the severity of the actual use of the aircraft. For this, a more reliable metric must be defined. The load cycle scale is a metric which more accurately reflects actual accumulated component strain.

Load cycles are calculated using reliable metrics such as torque, engine temperature and rotor speed. The HUMS then keep accumulative counters for the components for which load cycles are used, and alerts the maintenance crew whenever a component is about to reach its safe life limit. Load cycles must be calculated on all aircraft regardless of whether a HUMS is installed

or not. For aircraft with no HUMS, this task must be performed by other systems or by manual calculation.

Engine Power Assurance Check

Engine performance is gradually degraded throughout the lifetime of the engine. Performance must however not be allowed to drop below a certain minimum threshold. To ensure this, the engine Power Assurance Check (PAC) is performed at regular intervals, calculating the performance of each engine. The PAC consists in measuring the exhaust temperature needed to produce a given torque. On rotorcraft not equipped with HUMS, this procedure must be performed with engines running on the ground, using temporarily installed equipment.

2.3.3 Health Monitoring

The reactive part of a HUMS consists in detecting faults in the drive train as they occur, but before they become critical. This is a challenging task, as the system must be able to detect early in the propagation process, while at the same time not generate unjustified alarms.

Engine Vibration Monitoring

During engine power up and stabilized speed, temperature and vibration levels must be within certain limits defined by the engine manufacturer. These levels must be monitored at regular intervals to maintain airworthiness. For most HUMS, this task is performed automatically at each engine startup. On rotorcraft not equipped with HUMS, this procedure must be performed on the ground, temporarily installed equipment.

Transmission Monitoring

The health monitoring function tries to capture component condition using accelerometers mounted on the engines, gearboxes and shaft bearings. Chip detectors are also used on engines and gearboxes. A chip detector is capable of detecting metal debris in the lubrication. All rotorcraft are equipped with chip detectors generating cockpit warnings.

Rotor Track and Balance

In order to avoid violent vibrations at once per revolution of the main and tail rotor, the rotors must be well balanced. In addition, the track of each

blade must be adjusted, relative to the mast. Balance adjustments are made by adding or removing weights in the blades. Track is adjusted by changing the blade angle and profile. The vibration recordings required to calculate these adjustments are acquired during normal flight.

On rotorcraft not equipped with HUMS, this procedure must be performed with rotors running on the ground, temporarily installed equipment. Test flights are also required, to validate the result.

2.4 Impact of Current Technology

2.4.1 Reliability

Although effective in capturing several drive train failure modes, all existing HUM Systems are also responsible for generating a substantial number of unjustified warnings. The total number of warnings is aircraft and system dependent, but is reported by the operators [11] to be somewhere between 4.5 and 12 pr. 1000 flight hours, as a global average. The number of justified alerts is typically in the order of 1 or 2 pr. 1000 flight hours. Obviously, this number of false warnings can be quite overwhelming for inexperienced operators and the cause of a frustration for both HUMS personnel and management. Also, this creates a significant unregulated void in the procedures of rotorcraft operations.

All aspects aircraft operations are highly regulated. What maintenance work to perform, when to perform it, how to perform it, and what information to report to regulatory bodies and Original Equipment Manufacturer (OEM) is defined in fine detail. This applies of course also to any fly / no-fly decision, based on the outcome of maintenance inspections. The practical use of HUMS as a maintenance tool is however somewhat in contrast to this level of regulation.

Operators in the UK are obliged to submit documentation of their HUMS organizational structure and handling procedures to the CAA. Be that as it might, the day-to-day use of HUMS still leave waste room for subjective interpretation when it comes to HUMS based decision making. Even though the Eurocopter endorsed systems display reference to working cards in response to HUMS alarms, these can not be followed blindly. Obviously, a false alarm rate in the order of 4-1 would generate an immense amount of added (and unnecessary) maintenance work, if the alarms / working cards were to be followed without question. This leaves important decision making to the line technician or in best case to the company HUMS expert. As there is no formal training or certification for the interpretation of HUMS output,

it is up to each operator to maintain a level of training which ensures that safety is maintained. Thus, there are in reality no formal procedures for HUMS based decision making.

The Norne accident in 1997 did highlight the need for regulation of HUMS. In the Norne case, the aircraft was fitted with HUMS, but the sensor adjacent to the failed component was unserviceable at the time of the accident. If the HUMS would have been able to detect the fault, given a serviceable sensor, has been subject to debate. Regardless, the accident displayed the need for formal HUMS procedures and regulations, and was probably one of the contributing factors in the mandatory introduction of HUMS in the UK [1]. However, the regulations which are defined concerning HUMS address only the functionality and availability of the system. It does not specify formal procedures in the decision making process between HUMS output and possible maintenance responses.

In some cases, like the Eurocopter endorsed systems, the aircraft OEM and the HUMS provider is the same party. In these cases, the OEM can provide maintenance recommendations in cases where the operator is in doubt. However, the customer support throughput is usually not sufficient to provide diagnoses on flight-to-flight bases. As HUMS output should indeed to be analyzed between each flight, this still leaves much of the decision making to the line personnel.

There are no formal procedures for reporting detections and non-detections. As a result, it is difficult to create accurate statistics to determine which HUMS functions work and which do not. Some feedback is provided by the operators, but this information is highly biased and inconsistent. The following sections tries to extract whatever information possible, based on recorded data and expert opinions.

2.4.2 Safety

Helicopter accident rates have shown a clear downwards trend from the beginning of the eighties. Several measures, among them HUMS, where taken during the eighties to improve safety. Although it is difficult to quantify the effect of each measure, the safety enhancing effect of HUMS is none the less significant. The report "Helicopter Safety Study 2" by Sintef, states that HUMS is "the most significant isolated safety improvement measure during the last decade". The CAA estimates that about 70% percent of all drive train faults are uncovered by the current generation HUMS [38]. This figure is equivalent to the detection statistics available for the Eurocopter endorsed systems.

Despite good diagnostics capabilities for a wide range of failure modes,

several in-service difficulties have been reported by the operators. Some of these difficulties are related to the fault diagnosis technology available. Others are related to more practical usability issues which were not foreseen during the design of these systems.

2.4.3 Maintenance Credit

Changes in maintenance procedures, removal of maintenance tasks, or extension of component time between overhaul (TBO) due to the introduction of alternative monitoring techniques are referred to as maintenance credits. Maintenance credits to HUMS have been granted to the following functions:

- Load Cycle Calculation
- Exceedance Monitoring
- Power Assurance Check (PAC)
- Rotor Track and Balance (RTB)
- Engine Vibration Monitoring (EVM)

The functions listed above are mandatory functions on most helicopters. The calculation of usage cycles on non HUMS rotorcraft is performed by another permanently installed device. On HUMS rotorcraft, this function is simply embedded into the HUMS. In the case of PAC, RTB and EVM, HUMS is certified to replace temporarily installed equipment, used at fixed intervals. Performing these tasks on non HUMS rotorcraft require ground-runs of engines and / or rotors. In the case of RTB, test flights are also required. On HUMS rotorcraft, the information needed for tasks is recorded during the normal operation of the helicopter. This is clearly a cost saver, both in terms of maintenance man hours and even pilot man hours (for RTB technical flights).

Although an effective cost saver in some areas, HUMS contribution to reduced TBM is a different matter. As mentioned in previous chapters, the probability of a technical failure in rotorcrafts is minimized through regulation. The consequences of system fault in a given component is put in one of the following categories; Catastrophic, Hazardous / Severe, Major or Minor. The probability of component failure must be no greater than 10^{-9}, 10^{-6} or 10^{-3} pr. flight hour for the three upper categories respectively. For the rotorcraft transmission system, most components fall into the two upper categories. This means that a HUMS function set to monitor a component

which is "only" of Hazardous / Severe criticality must still have a probability of failure less than 10^{-6} / Flight Hour. This is a long way from the average detection rate of 70% experienced with the current systems. Although some of the diagnostic functions are well above 70%, there are still large regulatory boundaries which must be overcome on order to have any credit granted.

A major cost-driver in avionics development is the problem of hardware and software certification. A HUMS system which is to be qualified to Hazardous / Severe for a given function, must have airborne software certified in accordance to DO - 178 B Level B, which in itself is a feasible task. However, system criticality assessments are performed end-to-end. For instance, if a fault is captured by the airborne segment, but lost at the ground station due to buggy software, safety is obviously not maintained. For a Hazardous / Severe certified HUMS, this translates into level B software also on the ground station. As no operating systems are certified above level D, the entire ground station software, including operating system and hardware drivers, must be built from scratch. Further, all this software must also be certified to level B, which is a very expensive and time consuming task for such a large amount of software.

In theory, some mitigating solutions can be made to avoid this problem. This can for instance be to develop the software for two different platforms (OS + HW), and show that both solutions create identical results. Unfortunately, the Federal Aviation Authority (FAA) does not allow Commercial Off The Shelf (COTS) solutions containing software below level B in these cases. This means that custom made hardware must be ordered and certified for the ground station. Such a procedure would probably be even more costly than a level B software solution.

Given some improvements in detection reliability, HUMS has in theory a clear potential in the reduction of TBM. It is however difficult to see how any progress can be gained under the current regulatory regimes.

2.5 Objectives

The focus for this study is identifying methods which will improve fault detection rates and reduce false alarm rates for the health monitoring functions of EuroARMS and M'ARMS, two commercially available HUMS implementations manufactured by Eurocopter. An additional objective is to increase the autonomy of these solutions, so that they require little or no configuration by the user. The main axis of research is improving the fault detection methods which are based on vibration monitoring. Other sensor technologies for detecting propagating damage will also be discussed briefly. Further, the

Information Technology (IT) solutions providing the infrastructure for the health monitoring functions will be reviewed, and improvement recommendations will be made to avoid IT related problems becoming a limited factor for the performance of the system.

Improved prognosis based on more precise load cycle calculation is currently an important area of research. This path will however not be perused by this study. Nor will it treat problems related to airborne hardware, such as sensor and harness susceptibility to damage, digital hardware obsolescence, or practical problems related to the implementation of established usage monitoring techniques.

All tools used in this study, such as wavelets, artificial neural networks, and programming models are used without introduction. For any details on these technologies, the user might refer to the appendices and references.

Chapter 3

Current and Emerging Technologies

3.1 Introduction

This chapter explains the technologies that make up a HUMS. The state of the art for these technologies is reviewed, including an review of existing commercial solutions. From this, shortfalls for complying with the objectives of this study are identified. Finally, improvement potential for the existing solutions are derived, and a number of research areas recommended.

The HUMS diagnosis logic accepts a set of sensor signals and produces a diagnosis of the underlying assets based on this information. This requires a set of formal steps, including contextual validation and correction, feature extraction, and classification (Fig. 3.1). Contextual validation and correction is necessary in order to ensure that the data is representative for the state of the underlying assets. Any invalid data, like overly noisy data or data recorded in unfavorable conditions are removed or corrected at this stage. Such correction can be performed both before and after the feature extraction.

Feature extraction is to extract metrics about the system input which is more informative the evaluating at the raw input itself. The purpose of this step is to extract the essential characteristics of this input, so that it is more easily interpretable for the classifier. The classifier, for instance a fuzzy logic system or a neural network, is responsible for translating a set of features to an output diagnosis. As a classifier is no more than a mapping tool, its performance is no more consistent than the features presented to it. It is thus vital that the pre-processing steps, contextual correction and feature extraction, does a good job in extraction features which makes it easy to

distinguish the different classes, i.e. states of the underlying assets, that the classifier is supposed to recognize.

A classifier can be implemented as a neural network, fuzzy logic system, or simply a threshold tester. The classifier accepts the data generated by the feature extractor, and makes a decision on the state of the monitored asset based on this. As a minimum, the classifier must be able to distinguish assets in a normal condition from those behaving abnormally. In a more complex setting, the classifier can produce more detailed information such as fault recognition and expected time to failure.

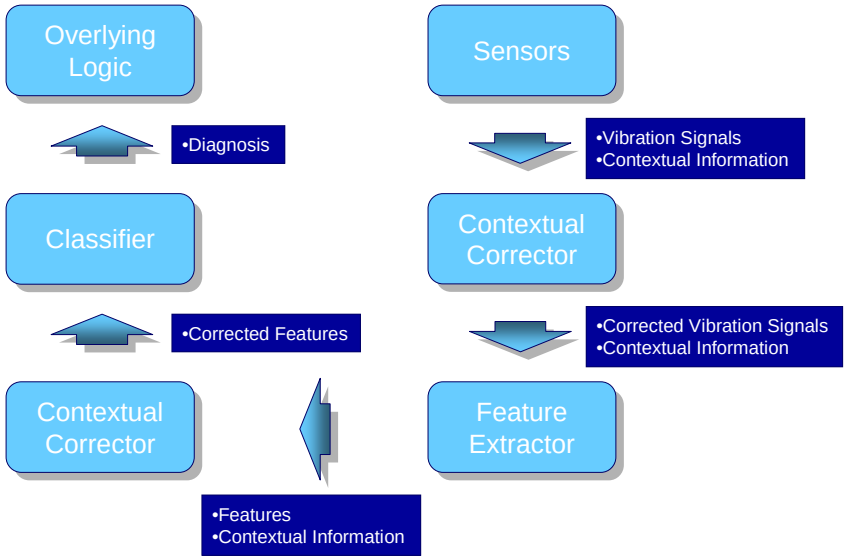

Figure 3.1: Diagnosis Overview

3.2 Data Validation and Correction

It is of course possible to test a mechanical assembly in a test-rig using a static environmental context, i.e. constant torque, constant rotation speed, constant temperature, and so on. A helicopter must however sustain substantial variations in operating conditions. The vibrations signature of all components is to some extent sensitive to variations in environmental context. Consequently, such variations must be compensated for before data is passed on to the classifier.

Obviously, any change in rotating speed for a mechanical assembly will change its vibratory signature. Even though the rotating speed of a helicopter drive-train is relatively constant, any variations which might occur must be compensated for. Further, the vibratory signature for some components is also susceptible to other contextual factors, such as torque. It is indeed of interest to compensate for such factors as well, so that the information passed on to the classifier is as consistent as possible.

3.2.1 Correction of Speed Variations

The vibration signature of a component is a function of its rotating speed. A gear will generate a tone, known as the meshing tone, at the frequency corresponding to the tooth pass frequency. The frequency of this tone, measured in Hertz, is obviously dependent on rotating speed. To uncouple rotating speed and vibration signature, the signal is re-sampled using synchronous sampling. Synchronous sampling means that the sampling interval is synchronous with the shaft rotation rather than time. Consequently, the resulting output has a fixed number of samples per shaft rotation rather than per second.

Synchronous averaging [43] refers to the process of recording a given number of rotations of a component, re-sample the signal to synchronize it with the shaft rotating speed, and adding together each segment representing one complete rotation. This will amplify any signal being periodic with the shaft rotating speed, and attenuate everything else. Synchronous averaging is a convenient tool for removing background noise. This is especially effective for gearboxes, where a single accelerometer will capture the vibration signatures of several components. By creating a re-sampled and averaged signal for each component, each resulting signal contains the vibration signature from only a single component. A few cases do however exist, where a signal captures the signals from several components. These are the cases where two similar components, like two gears or two bearings, are located in close proximity rotate at the same speed. In these cases, only the applicable selection of vibration features can separate the characteristics of each component. For some components, like the epicyclical planet gears, also the same vibration features are applicable for each component in the acquisition. Thus, no unambiguous error localization can be made.

Synchronous averaging is typically used for shafts and gears. Bearing acquisitions are typically re-sampled, but not averaged. This because roller slip will cause a phase delay in the vibration signal, causing it no longer to be periodic with the shaft rotation.

3.2.2 General Contextual Correction

Although the vibration signature from all rotating components is sensitive to rotating speed, some vibration signatures are also sensitive to other factors. Helicopters in normal use experiences a large variation in contextual parameters, such as altitude, speed, oil temperature, torque, etc. Torque is a well known influence especially on gears.

Because the environmental context is random in time, variations in environmental context are manifested as random variations on the recorded vibration signals, and consequently the vibrations features. Most commercial HUMS amend this problem by using a contextual window in where acquisition is allowed. This involves setting maximum and minimum thresholds for key parameters, such as speed and torque. A drawback of this method is that the contextual variation within the window can be substantial. Reducing windows size might reduce random variation, but risk reducing the data volume collected.

A supplementary method is by using a model representing the influence of contextual variations on the different vibration features. Once models are estimated for each feature, they can be used to cancel the effect of contextual variations. This method has been successfully deployed using engine torque as the only environmental context [21].

3.2.3 Epicyclic Frequency Separation

Frequency separation is a pre-processing technique particular to epicyclic planet gears and bearings. An accelerometer monitoring an epicyclic gear stage must, for practical reasons, be placed outside the gearbox housing. This means that the accelerometer will pick up the vibration signatures of the ring gear, the sun gear and bearing, as well as all planet gears and bearings. The ring, sun and planet vibration signatures can easily be separated using synchronous averaging, as these components rotate at different speeds. This method will however not separate the different planet signatures, as all planet gears and bearing are rotating at the same speed. Consequently, it is not possible to pinpoint any detected planet fault to a specific planet gear or bearing. Further, the error-indicating features from one faulty gear or bearing will get buried in the normal state vibration signatures from the other planets, making fault detection difficult.

A method known as frequency separation [32] [31] was developed to amend this problem. Frequency separation method requires an indexer to be placed on the planet carrier, so that it is possible to know when each planet passes the accelerometers. The recorded signal is then split up into equal

size windows, where the number of windows equals the number of carrier rotations time the number of planets. Phase is adjusted so that each window contains one planet passing the accelerometer. The windows are then sorted by planet, forming one new signal for each planet.

3.3 Feature Extraction

Feature extraction is the process of extracting metrics about the system input which are more informative than evaluating at the raw input itself. Input features are the meta of the input, and constitutes a higher order interpretation. Feature extraction is a parameterization process which often reduces the data volume, though this is not always the case. Desirable properties for features are that they are sensitive to the characteristics of the input which differs between classes, while insensitive to characteristics which differ within each class. The latter typically being insensitivity to measurement noise and other irrelevant factors which might confuse the classifier.

In the case of vibration monitoring, a brute-force approach to feature extraction is extracting the Discrete Fourier Transform (DFT) of the vibration signal. The absolute value of the DFT contains an estimate of the signal power spectrum, which displays substantially different behavior between health state and damaged state signals. Further, the absolute DFT is insensitive to the shaft phase offset, which is random and thus a source of variation in signal characteristics within each class.

Given the geometry of a mechanical assembly, it is however possible to predict which frequencies, i.e. DFT coefficients, are affected by different failure modes. Consequently, any other coefficient becomes less relevant. Further, some fault-indicating signal characteristics are not well captured by the DFT, but are better enhanced using other transforms. Thus, it is common to design feature extractors which outputs only the information relevant for detecting the failure modes to which the associated components are susceptible. This information are in the context of HUMS referred to as indicators.

3.3.1 Condition Indicators

The feature extraction part of a HUMS attempts to isolate signal features which have substantially different behavior in normal state signals and signals recorded from damaged components. For shafts and bearing, this process is fairly straight forward. Normal state shafts do not produce much vibration energy. Shaft failures, such as unbalance and miss-alignment, are easily

identifiable as vibration energy increases at the frequencies corresponding to multiples of the shaft rotation frequency. Classical bearing failures are, as already explained, identifiable as periodic energy pulses with frequency given by the rotation speed and bearing geometry, as well the fault type.

For gears, feature extraction is not that simple. According to [30], a perfect gear produces a distinct meshing tone (Eq. 3.1), with a harmonic distribution P_n given by the geometry of the gear, over a noise floor $w(n)$. The variables z, Ω and Φ_n symbolized shaft rotation frequency, the number of gear teeth, and phase offset for each harmonic.

$$x_{perfect}(t) = \sum_{n=0}^{\infty} P_n cos(ntz\Omega + \Phi_n) + w(t) \tag{3.1}$$

Due to the imperfect nature of any physically gear implementation, each gear mesh harmonic is subject to amplitude and phase modulation by any multiple of the shaft rotating frequency (Eq. 3.2).

$$x_{realistic}(t) = \sum_{n=0}^{\infty} a_n(t) cos(ntz\Omega + b_n(t)) + w(t) \tag{3.2}$$

$$a_n(t) = \sum_{k=0}^{\infty} A_{k,n} cos(nt\Omega + \alpha_{k,n}) \tag{3.3}$$

$$b_n(t) = \sum_{k=0}^{\infty} B_{k,n} cos(nt\Omega + \beta_{k,n}) \tag{3.4}$$

Consequently, a gear vibration signature becomes a function of the amplitude modulation amplitude matrix $A_{k,n}$, the amplitude modulation phase matrix $\alpha_{k,n}$, the phase modulation amplitude matrix $B_{k,n}$, and the phase modulation phase matrix $\beta_{k,n}$. As the coefficient values tend to drop off quickly for increasing values of n and k, simplified finite-size approximations of these matrices can provide a good approximation of a gear vibration signature.

According to [42], any presence of gear failures tends to increase the modulation between the meshing tone harmonics and low multiples of the shaft rotation. This corresponds to a value increase in the coefficient matrix $A_{k,n}$ for low values of k. Traditional condition indicators are designed to capture this phenomenon. Indicators do also exist which capture changes in the noise floor $w(t)$, which also is associated with certain types of damage.

Overview

The indicator definitions presented here assume that the input signal is finite, which is the case for all commercial HUMS. It is indeed possible to create indicator algorithms working on infinite signals, but this topic is not treated in this study due to lack of relevance in the context of HUMS. The indicators explained here are only few examples of the total number existing in the literature, and only an extract of those are given an in-depth explanation.

Indicator	Damage Detected	Ref
IR	Bearing inner race crack	[35]
OR	Bearing outer race crack	[35]
BS	Bearing roller crack	[35]
Crest Factor	General gear	[10]
Energy Operator	Localized gear	[26]
Energy Ratio	General gear	[44]
FM0	General gear	[42]
FM4	Localized gear	[42]
Kurtosis	Localized gear / bearing	[39]
M6A	Localized gear / bearing	[28]
M6A*	Localized gear / bearing	[44]
M8A	Localized gear / bearing	[28]
M8A*	Localized gear / bearing	[44]
MOD	Gear web crack	[42]
NA4	Localized gear	[55]
NA4*	Localized gear	[17]
NB4	Localized gear	[53]
NB4*	Localized gear	[54]
RMS	General	[10]
RMSR	General gear	[44]
Ω_1	Shaft unbalance	
Ω_2	Shaft misalignment	
Ω_{zn}	General gear	

Table 3.1: Common condition indicators.

Root Mean Square

The root mean square represents the energy of the signal. As most serious defects in gear and bearing assemblies will increase the signal energy, this is a general fault indicator.

$$RMS_x = \sqrt{\frac{1}{N}\sum_{n\in N}(x(n)-\mu_x)^2} \tag{3.5}$$

$$\mu_x = \frac{1}{N}\sum_{n\in N} x(n) \tag{3.6}$$

Residual Energy

The residual signal [55] is given by (Eq. 3.7), where DFC_x is the DFT coefficients of x. This transform captures the noise floor $w(t)$ of the signal, by removing the signal components corresponding to the harmonics of the meshing tone. An alternative definition [42] exists, which also removes the signal components corresponding to the first modulation sidebands. By calculating the rms of the residual signal, $RMS_{x_{res}}$, the energy of the signal noise floor is estimated. Several gear failures tend to increase the noise floor, making this an indicator both to localized and distributed damage.

$$x_{res} = x - DFT^{-1}[MDFC] \tag{3.7}$$

$$MDFC_k = DFC_k.[modulus(z,k)! = 0] \tag{3.8}$$

Residual Energy Ratio

The residual energy ratio is the ratio between the residual energy and the total signal energy. Alternatively, it can be defined as the ratio between the residual energy and the meshing energy [44]. The former definition is always between zero and one, where zero indicates the perfect gear definition (Eq. 3.1).

$$ER = \frac{RMS_x}{RMS_{x_{res}}} \tag{3.9}$$

Kurtosis

Kurtosis is the fourth statical moment of a dataset, and indicates how outlier-prone the dataset is. In vibration monitoring, this provides a good shock indicator, indicating if a small portion of the signal has significantly higher amplitude than the rest. Kurtosis is associated with localized gear damage, as well as a cracks and corrosion for bearings.

$$Kurtosis_x = \frac{\sum_{n \in N} (x(n) - \mu_x)^4}{RMS_x} \tag{3.10}$$

Omega

With Ω being the shaft rotation frequency, the Ω_n is simply a spectral pointer defined relative to the shaft rotation. For synchronously sampled signals, Ω_n corresponds simply to the n'th DFT coefficient. The values 1 and 2 for n, denotes frequencies for detection of shaft unbalance and misalignment respectively. Values for n being multiples of the number of teeth extracts frequencies associated with gear damage.

Modulation

According to (Eq. 3.1), a perfect gear should only produce vibration energy at multiples of its tooth pass frequency. A gear hub crack will however create a different energy of the meshing tone depending on the rotational position of the gear. Thus, gear rotation and meshing becomes modulated. This will manifest itself as modulation sidebands to the harmonics of the meshing tone, with sideband distance to the carrier equal to the shaft rotation frequency. Monitoring these frequencies will provide indications of gear web cracks, severe localized damage, and unbalance in the gear shaft [42].

Bearing Indicators

A crack in the inner race or outer race of a bearing will manifest itself as a pulse repeated every time a roller passes over the crack. A crack directly on the roller will generate a pulse every time the crack passes one of the races, i.e. twice for every rotation of the roller. This gives the three fault frequencies of a bearing; ball pass frequency inner race (IR) ball pass frequency outer race (OR) and ball spin frequency (BF) [35]. These frequencies, relative to the shaft rotation, are specific to each bearing.

Monitoring any of these frequencies directly will however not detect any faults, as repeated pulses on these frequencies will become modulate on the natural frequency of the bearing, and end up as sidebands to this frequency. As the natural frequency normally is high, and not necessarily known, looking for modulation sidebands in the expected locations is not practical.

A better approach is to demodulate the signal. The signal envelope, or Hilbert transform, will demodulate the bearing fault frequencies from the carrier and project them back to their expected locations. Calculating the DFT of the enveloped signal will thus reveal any bearing damage. Normally,

an area of $\pm 10\%$ around each fault frequency is extracted to accomodate for roller slip.

3.3.2 Stationarity Indicators

Although the basic condition indicators provide reliable indications to change in the condition in the underlying assets, they are of little use without a comparative baseline. Rather than defining a baseline for each indicator, it is possible to compare each observation with the most recent ones to look for any trends in the evolution of the indicators. A simple method is to perform a linear regression of the last couple of observations, and measure the rate of incline or decline over this segment [33] [21] [22]. Alternative, it is possible to extrapolate the linear model, and estimate the time remaining before it crosses some pre-defined threshold. If a condition indicator is seen as a parameterization of the raw sensor signal, a stationarity indicator constitutes a second level parametrization.

3.3.3 Modeling

A more general approach to feature extraction is modeling. A modeling approach does not, unlike traditional condition indicators, make any assumptions about features of importance, and does not require any a priori information about the geometry of the underlying assets.

General parametric signal models are MA, AR and ARMA [4]. By assuming that a signal power spectrum is stationary, this power spectrum can be approximated by any of these models. Fitting a model to an observed signal can be done by a number of algorithms found in the literature [36]. The number of parameters for any of these models fitted to an observed signal are far subsiding the number of DFT coefficients for the same signal. Consequently, these parameters make a set of features suitable for classifier input. This was successfully tested in [14] [20], using a cluster classifier.

A similar approach is using the lifting scheme [45] to generate a wavelet capable of predicting a signal waveform. This method involves deriving a wavelet from a normal state transmission. The same wavelet can then be used for time domain prediction of subsequent observed signals. Any substantial prediction error indicates that the observed signal does not correspond to the normal state wavelet, and is thus an indication of failure [7] [40] [41].

3.4 Classification

With the exception of the usage functions, which utilize simple and precise metrics for decision making, HUMS lies within the field of pattern recognition. There are however a few characteristics which separate HUM Systems from most other pattern recognition systems. This is mainly due to the criticality of detecting all failure modes, regardless of their frequency of occurrence. Consequently, the systems are set to detect failure modes for which they are not trained, even some of which have never even occurred (and maybe never will). It is to some extent possible to extrapolate the tested and confirmed diagnosis functions of one component to other components for which training data does not exist. This is however not done without adding even more uncertainty to discipline which by default is quite "fuzzy", and is partially the reason for the high false alarm rate experienced with these systems.

3.4.1 Threshold Testing

Condition indicator threshold testing is the oldest classification technique in the HUMS field, and is incorporated in several commercially available solutions. The technique consists simply of testing each indicator to a threshold (Fig. 3.2). Given the type of indicator and the component from which it originates, at threshold breach gives both an indication that something is wrong, as well as information on which component is faulty and what type of failure it suffers from. In a practical implementation, it is common to require N out of M threshold overshoots on a given indicator before an alarm is raised. This is to avoid that indicator outliers, in the context of HUMS referred to as spikes, result in unjustified alarms.

The main objection to threshold testing in health monitoring is the difficulty in setting the optimal threshold values. Setting thresholds too low might result in false alarms, i.e. threshold overshoots despite the fact that nothing is wrong. Setting the thresholds too high renders the system less sensitive to variations in the vibration signature, and thus less equipped for detecting faults. For some indicators, it is possible to set global or fixed thresholds. This means that the same threshold is applied across an entire fleet. Unfortunately, most indicators have a normal state envelope which is unique to each aircraft. Further, this envelope is prone to change between major overhauls, a phenomenon known as a step change. To accommodate for this, thresholds must constantly be updated for each aircraft.

Threshold adjustment, or learning, is performed on new aircrafts and after major overhauls. The process consists in acquiring a statistically significant

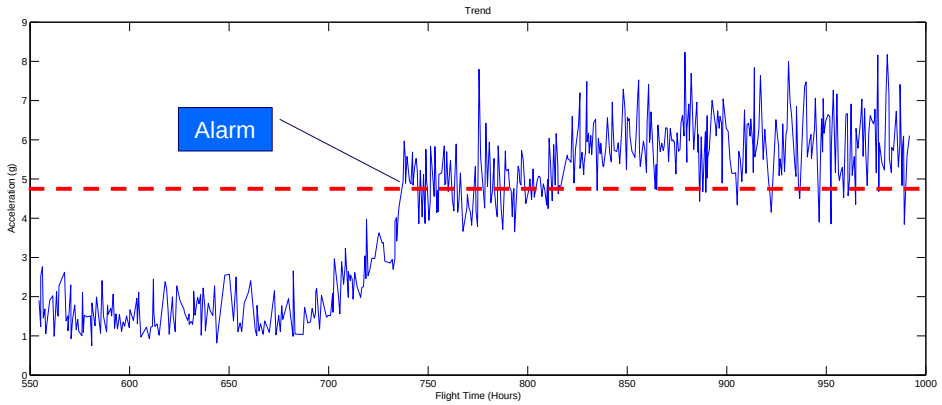

Figure 3.2: An indicator breaching its threshold.

baseline of observations, typically on the magnitude of 50 flight hours, and calculating the gaussian localization μ_i and distribution σ_i parameters on the dataset. The threshold or thresholds for an indicator i are then defined using a threshold policy of type $T_i = \mu_i + N\sigma_i$. During the learning period, a set of alternate thresholds are used. These are global, and are to avoid false alarms set so high that they have reduced chance of detecting faults. Consequently, the aircraft is vulnerable during the training period.

Threshold re-learning is a tedious task for heavy aircraft with several hundred indicators, and it is not always possible to predict which overhauls will require re-learning of which indicators. This burden is a common complaint from operators who wishes more autonomous solutions.

Alternative variants are hysteresis thresholds, hypothesis testing and Bayesian decision approaches. Hysteresis thresholds are applicable in systems where it is necessary to measure the number of times a variable crosses a threshold over a given period. This method is used in several of the usage monitoring functions of the HUMS, but has no obvious applications in health monitoring.

Using hypothesis testing it is possible to compare two groups of observations, and find the possibility of the two groups originating from the same distribution. If one group represents the normal state baseline and the other a set of observations from an asset in an unknown condition, it reasonable to assume that the asset is in a damaged state if its associated observation distribution is highly different from the normal state baseline. This is in reality a generalization of the threshold testing method described above, but permits comparing a group of samples to the learnt baseline. Another variant is analyzing the possibility of various failure modes given an alarm. By knowing these prior probabilities, it is possible to identify the most likely problem,

given a series of alarms. This has successfully been applied to rotorcraft condition indicators in [37].

The main drawback with the two latter methods is that they require a substantial amount of observations in order to produce a diagnosis. This means that there will be a delay between the occurrence of a problem and its detection. As far as Bayesian decision making is concerned, it is due to limited availability of training data difficult to estimate the prior possibilities.

3.4.2 Clustering

Most failure modes tend to affect more than one indicator. A gradual shift in several indicators is thus a better indication of failure than random perturbations in a single indicator. Consequently, a more robust indication of failure is measuring the total drift across all indicators for a given component, relative to their normal state baselines.

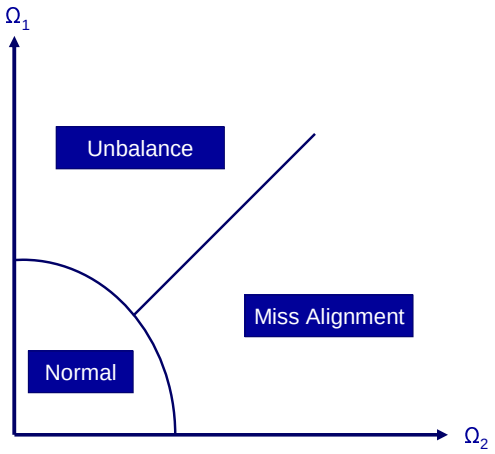

Figure 3.3: Relevant clusters for shaft fault detection.

A classifier taking this into account can be implemented through a cluster system [16] [18]. A cluster system is a space with a number of dimensions equal to the number of inputs. Each class is a multi-dimensional region in this space. Any input vector is classified by determining which sphere it falls within (Fig. 3.3), or alternatively classified as unknown it falls in the void between the regions.

A HUMS classifier based on this method must as a minimum implement the normal state class. Consequently, any observation falling outside this sphere must be considered faulty. Such a system might also implement classes

representing known failure modes. This technology has been adapted for several commercial solutions [3] [22], and provide a classification tool which is both flexible and transparent.

3.4.3 Feedforward Networks and Fuzzy Logic

Certain solutions based on feedforward networks and fuzzy logic exist in the academic literature [39] [25]. Compared to cluster solutions, the feedforward network is more efficient by allowing complex class regions to be defined with fewer neurons. The principal objection against feedforward networks working directly on DFTs or condition indicators is that training requires non-linear optimization. Using a non-linear optimization in the learning process will result in an even more complicated post-overhaul re-learning procedure for the operator. This can be circumvented by normalizing the inputs against learnt baselines before entering them into a factory-set network, although this solution has not been addressed in the literature. Moreover, feedforward networks require substantial amounts of training data and provide less insight to their inner logic. It should also be noted that flexibility similar to a feed forward network can be achieved with a cluster solution by adding a linear layer behind the radial basis layer.

It is a well known phenomenon in vibration monitoring that damage to one component can cause perturbations in the condition indicators of adjacent components. This might cause confusion as to what the actual fault is, and where it originates from. Fuzzy logic solutions tying together the individual classification systems of components into system-wide or aircraft-wide diagnosis has been suggested to amend this [24] [13]. The physics involved in inter-component vibration propagation are however quite complex, and not always well understood by the OEM. Moreover, it is difficult to gather sufficient relevant training data to create and validate a robust solution.

3.4.4 Prognosis

Threshold testing can in combination with trend analysis be used for prognosis. This is the case both for single indicator thresholds, as used with traditional threshold testing, and multidimensional thresholds, as used with clustering systems. Given the value and the slope of an indicator progression at a point in time, it is possible to estimate the time remaining until the threshold will be breached. In a clustering system, this corresponds to the estimated time remaining until the system leaves the normal state cluster, given the current position and gradient. This estimate obviously assuming that the progression slope or gradient remains constant.

For this to have a significant operational interest, the link between the threshold, or cluster, and the mechanical state it represents should be clear and well understood. If classifier training is based purely on statistical analysis of a normal state indicator distribution, the prognosis will simply give the estimated time until the vibration signature changes from normal to abnormal, given an arbitrary definition of normality and abnormality. As the mechanical state associated with the threshold or cluster border is unknown, the estimated time until the mechanical system reaches this point will have less operational interest.

3.5 Commercial Solutions

Several commercial HUMS implementations exist from several manufacturers. This section introduces a few of them. The selection of systems discussed here is however biased toward systems developed for heavy rotorcraft operating in hostile environments, as this was the origin of the HUMS development. Numerous implementations and manufacturers have since joined the HUMS marked, addressing both the original audience, as well as new markets such as military operators and medium or even light rotorcraft.

3.5.1 IHUMS

IHUMS is manufactured by Meggitt Avionics in cooperation with rotorcraft operator Bristow [2]. It is a first generation system, and was one of the first two systems on the market. IHUMS ground stations with graphical user interface exist for both UNIX and Microsoft Windows NT. Diagnosis is based on condition indicators processed by a fuzzy logic-like matrix system. The matrix system aids in suppressing spurious alerts due to indicator spikes.

3.5.2 North Sea HUMS

North Sea HUMS was developed by British avionics company SHL (currently Smiths Aerospace). It is also a first generation HUMS, and is together with IHUMS one of the first two systems on the market. Although in the process of becoming obsolete, it is still widely used and has a good reputation among its users. North Sea HUMS uses a ground-station running on UNIX, and offers remote desktop capabilities. Diagnosis is based on basic condition indicator thresholds.

3.5.3 EuroHUMS

In 1993, the Norwegian operator Helicopter Service negotiated HUMS installation as part of a Super Puma purchase contract. Eurocopter did however not have a HUMS program at the time. As an intermediate solution, HUMS development was outsourced to SHL. SHL were one of the most experienced companies in this field, and had already developed a HUMS for Super Puma, North Sea HUMS. EuroHUMS is simply a Eurocopter customized version of this system.

3.5.4 GenHUMS

GenHUMS is manufactured by Smiths Aerospace. The GenHUMS grounds station works standalone, but can also relay data back to the OEM. The system uses a cluster-based data fusion technique merging all indicators from a component into a single value. This data fusion indicator is then subjected to trend analysis, uncoupling aircraft specific and fault indicating features [22]. Although the trend analysis part is not yet commercialized, this constitutes one of the most interesting advances in HUMS research, as it makes it possible to produce an autonomous HUMS not requiring aircraft specific training.

3.5.5 IMD HUMS

IMD HUMS is manufactured by Goodrich Fuel Systems. This system features state of the art diagnosis methods, including flight regime recognition for more accurate load and wear estimation based on actual use. The IMD HUMS grounds station works standalone, but can also relay data back to the OEM.

The IMD HUMS compensate for torque variations using a "bucket"-based system, sorting indicator values into classes given by the torque at the time of the acquisition [29]. The data in each class is then processed individually. Further processing involves deploying a cluster-based data fusion technique merging all indicators from a component into a single value [3]. These values are normalized, based on training data, to stay between zero and one for any possible input. This makes it easy to mark up normal, suspect and faulty regions for each component.

3.5.6 T-HUMS

T-HUMS is manufactured by Israeli avionics company RSL. This is a military system which, contrary to civilian HUMS implementations, is capable of performing the bulk of its calculations in flight. It can also be fitted with a cockpit display providing the pilots with real-time battle damage assessment.

The T-HUMS uses condition indicators based on several transforms, including DFTs, Ceptrum and periodograms from non-averaged signals [21]. Indicators are then normalized against learnt baselines to representing the normal state aircraft specific vibration signature. The system then compensates for environmental changes using a polynomial approximation of the relationship between vibration signature and environmental conditions. It also applies trend analysis based on linear regression both on condition indicators, thus creating new indicators, and on classification results. This is done using two frame sizes, detecting long and short term tendency. The indicators are then subject to further processing by classification algorithms such as cluster systems, fuzzy logic, and artificial neural networks.

3.6 M'ARMS and EuroARMS

Eurocopter is currently supporting two HUMS; Modular Aircraft Recording and Monitoring System (M'ARMS), and its predecessor Eurocopter Aircraft Recording and Monitoring System (EuroARMS). Although EuroARMS and M'ARMS are two different systems, they inhibit the same functions. Thus, these two systems are in this report jointly referred to as Aircraft Recording and Monitoring System (ARMS). Both systems collect data while in operation, which are downloaded to a PCMCIA flash memory card after each flight (Fig. 3.4). The content of the flash card is analyzed at a Windows NT or Server 2003 workstation using specialized software. This workstation is referred to as the ground station.

3.6.1 Airborne Segment

The ARMS airborne segment taps into the Arinc databus, which is used for transmitting data between different system modules. This provides access to contextual information such as altitude, temperature and air speed. Contextual information is used for generating parameter threshold overshoot alarms and estimating component load cycles. Further, this information is used for determining if the aircraft is in a flight stage when it is possible to perform vibration acquisitions.

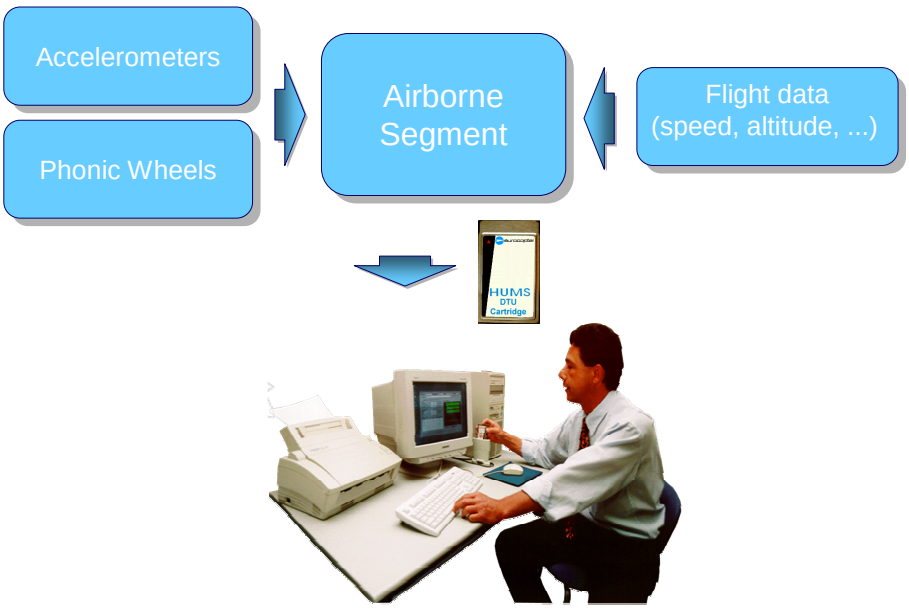

Figure 3.4: ARMS data flow

In addition to using data acquired through the Arinc bus, the ARMS has its own set of sensors. This includes speed sensors mounted on the engine compressors, engine turbines, main rotor and tail rotor, as well as a number of accelerometers. The number of accelerometers is aircraft specific, but does generally cover engines, all gearboxes, oil cooler, rotors and the tail drive shaft.

During one acquisition cycle, the system acquires finite length acquisitions from all monitored components, following a preset program. A total of six acquisition cycles are performed per flight; one on the ground, and five in cruise (Fig. 3.5). To save space, acquisitions are immediately re-sampled and averaged with the shaft rotation speed. This shortens gear, shaft and rotor acquisitions from 200 rotations to simply 1. Vibration signals, parameter exceedance alarms and load cycle calculations are stored on a data cartridge at the end of the flight. The cartridge is then analyzed at the ground station.

3.6.2 Ground Segment

The ground station performs a number of functions. Upon receiving the data cartridge from the aircraft, it generates a maintenance report containing all parameter exceedances encountered during the flight, accompanied by their corrective actions. It also keeps track of load cycles, and alerts the user if a

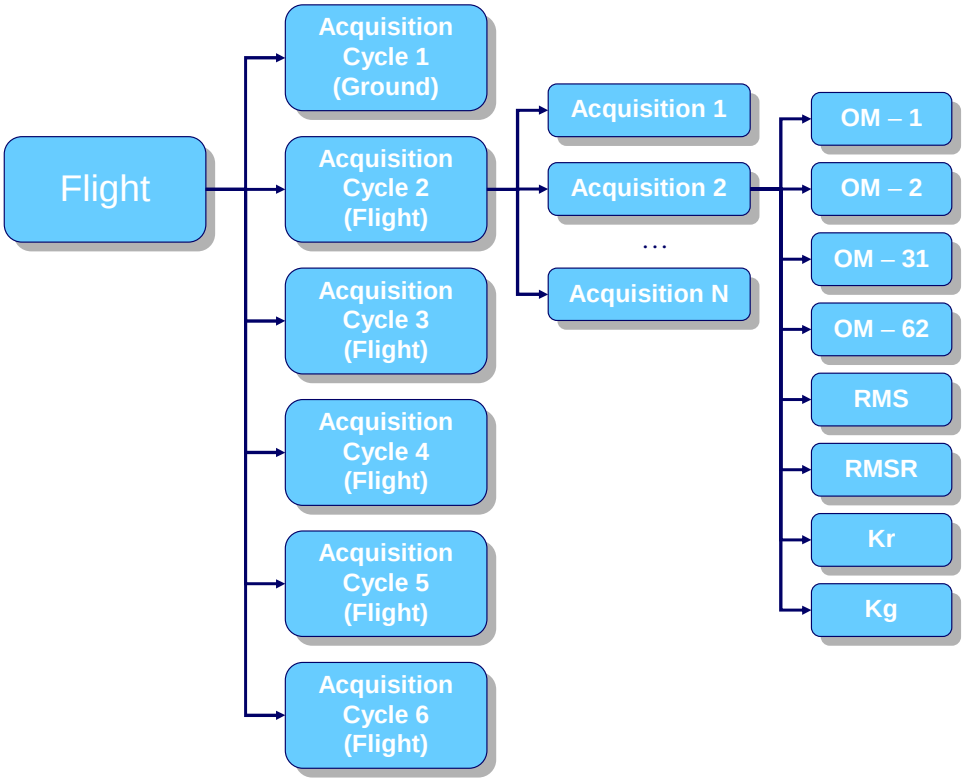

Figure 3.5: ARMS acquisition cycles

component has reached its safe life limit.

On the vibration acquisition of each component a number of indicators are calculated. These are then evaluated against thresholds to determine if a fault is present on the associated component. Indicator threshold breaches are added to the maintenance report, accompanied with the maintenance actions for the detected fault types.

While some thresholds are globally fixed, most are individual to each aircraft. These thresholds are set based on a training period, where the normal state location and dispersion for each indicator is identified. Following major overhauls, aircraft individual thresholds must be reset, as overhauls changes the vibration signature of the transmission system.

3.6.3 Decision Flow

Under normal circumstances, the operator is able to determine if a fault is present on a helicopter by evaluating the ground station output. When in doubt, the operator submits a defect report to Eurocopter customer support. This is normally done by fax / email, with screenshots of the affected indicator plots attached. An expert evaluation is then returned to the operator (Fig. 3.6).

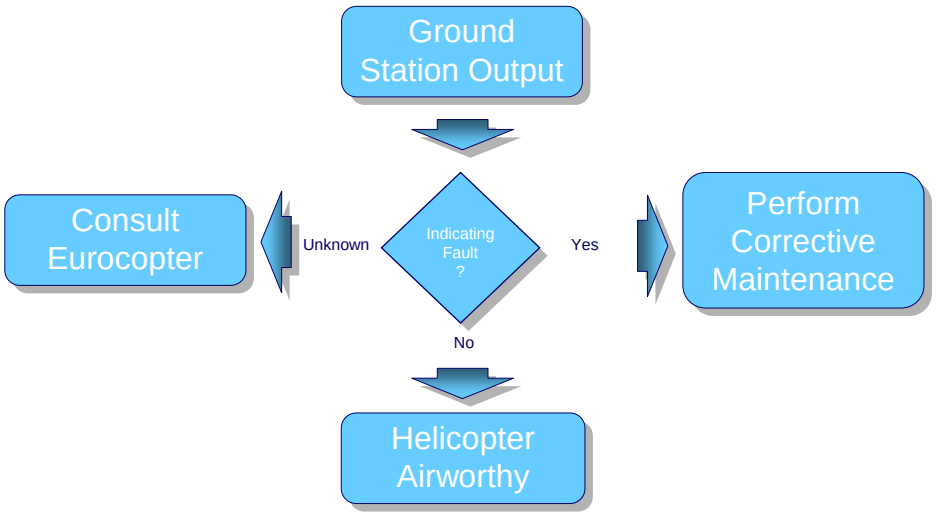

Figure 3.6: ARMS decision flow

In addition to this problem driven data flow, digital HUMS data is submitted to Eurocopter on more or less regular intervals. This is done by sending data backup tapes in the mail. This approach is however too slow to use in support cases, as the backup tapes can take several days before reaching Eurocopter.

3.6.4 Improvement Potential

This study limits itself to address surveillance of the transmission system between the engines and the rotors, as well as any IT-related problems related to this task. Monitoring of engines and rotors are the topic of other studies, and will therefore not be treated here. Any problems related to the implementation of traditional and well proven usage functions, like parameter exceedance warnings, are considered to be more a development / quality assurance nature, and are thus also excluded from this study.

This section suggests a number of possible improvements of the ARMS solutions. The overall aims for these propositions are to:

- Reduce false alarm rates
- Increase detection rates
- Reduce operator workload

Only methods related to vibration monitoring, or methods replacing existing vibration monitoring techniques are considered. Some of these propositions are outside the scope of this study, in the sense that they can not be implemented and tested during the assigned time-frame. They are none the less included for future reference, and as recommended topics for future studies.

Acquisition Rate

The systems currently in use perform a maximum of five acquisition cycles per flight. This choice was made due to hardware limitations at the time when the first systems were created. In addition, it was believed that vibration-based condition indicators could be reliably interpreted like any other flight parameter. Experience has however shown that condition indicators have considerable scatter, and are usually best regarded as signals themselves, subject to further signal processing and statistical analysis methods.

A way to increase system reliability is to increase the number of acquisitions per flight. Due to the false-alarm rate of existing HUMS, the probability of a fault being present on a component, given an alarm, is not very high. With repeated alarms, the probability of a fault being present will however quickly converge [37]. The more data is acquired per flight hour, the quicker, in terms of flight hours, a reliable diagnosis can be produced. This is provided that the recorded data is representative, and not polluted by contextual variations.

Evaluating data over a large number of acquisitions with current system does however pose a serious threat of overlooking rapidly propagating faults, due to the low acquisition rate of these solutions. Acquisition rate is however in part a limitation of the airborne hardware, and can not necessarily be altered by simply modifying software or configuration. Increased acquisition rate should however be kept in mind when designing the next generation airborne segment.

Sensor Fault Detection

It is well agreed that the most flawed component monitored by the HUMS is the HUMS itself [11]. Although the ARMS solutions possess self test functions capable of identifying problems like dysfunctional circuit boards, capabilities for robust detection of partially damaged sensors or harness are less developed. A consequence of this is that the changes in vibration signature caused by sensor and harness degradation are frequently interpreted as mechanical damage by the ground station. Efforts should therefore be made to develop indicators capable of distinguishing between electrical and mechanical problems.

Due to lack of relevant data, research in this area will require extensive test rig experiments.

Contextual Data Correction

The current implementations of Eurocopter Aircraft Recording and Monitoring System (EuroARMS) / Modular Aircraft Recording and Monitoring System (MARMS) use contextual validation to decide when acquisition is enabled. This is to limit the number of environmental contexts for when acquisition is performed, thus reducing indicator scatter. There are however still significant variations within this window, contributing to considerable variance between acquisitions.

Efforts should be made to investigating the correlation between different environmental contexts, flight stages, and vibration signatures. If such correlations can be modeled, these correlations can also be compensated for. Environmental normalization of condition indicators has already been proposed in [21]. There are however no methods in the literature which address the correction of vibrations signals. This is of interest for long-durations acquisitions typical for rotors and epicyclic carriers. In these cases, acquiring a single finite length signal takes several seconds. This leaves a considerable probability for the environmental context changing throughout the acquisition period. Consequently, the raw signal must be piecewise corrected before any indicators are calculated.

Implementing this method for EuroARMS / MARMS will however require redesign of the airborne segment. This because the method must be applied before synchronous signal averaging. Any modification to the airborne software is however highly expensive. Consequently, this solution should be kept in mind for the next evolution of the airborne segment.

Epicyclic Monitoring

Due to high complexity and several parts monitored by a single accelerometer, fault detection in epicyclic gearbox stages are inherently difficult. Apart from the problem of long distance between components and sensors, several components are captured in the same acquisition. This means that the error indicating signature from a faulty component will be buried in the normal state signatures from the healthy state components captured in the acquisition. A method is already proposed in [32] [31], which deals with this problem.

Like with contextual signal normalization, this method must be applied before synchronous signal averaging. Further, the method requires an indexer on each epicyclic stage, which is currently not available for gearboxes with multiple epicyclic stages. This need for modification of both airborne software and hardware makes this an expensive solution which should be considered for the next generation airborne segment.

New Sensor Technologies

The existing chip detector technology is reliable, but provides warning at a very late stage. For fault types such as gear fretting, the chip detector will provide warning only after severe damage. Obviously, the purpose of condition monitoring is to uncover faults at a much earlier stage. Gear fretting is also notoriously difficult to detect through vibration monitoring, because it produces little low frequency vibration. Oil debris monitoring differs from classical chip detection by providing precise quantitative and qualitative analysis of oil debris. This technology might prove to be quite effective for monitoring of gearboxes, as fretting tends to cause substantial amounts of fine grained metal debris in the lubrication.

A weak-spot for all HUM System in use today is detection of gear fretting and bearing corrosion. These failure modes typically create metal-to-metal contact, which is a generator of weak signals at high frequency. As the signals are quite weak compared to the low frequency vibration, in addition to being out of the sensitive spectrum of most accelerometers, they are easily lost. Further, the use of synchronous averaging on gears will efficiently suppress any signal not correlated with the shaft rotation. This includes the metal-to-metal noise created by fretting.

A method better suited for detecting these failure modes is Acoustic Emission (AE) monitoring. Acoustic emissions are ultrasonic energy emissions created in response to metal-to-metal contact and metal deformation. This information is normally recorded through acoustic sensors or wide band ac-

celerometers, in an asynchronous manner. AE monitoring systems are very sensitive to early signs of gear / bearing failure, mainly metal deformation and direct metal-to-metal contact. Thus, it should have the properties necessary to detect both bearing corrosion and gear fretting. In addition, this technology might be able to detect fretting between statically assembled components. Examples of problem areas are loosening of shaft splines, gearbox housing joints, and gear fastenings bolts. The latter is a known problem on the Super Puma LH ancillary gearbox, where the intermediate gear fastening bolts tend to loose torque. Being able to detect this phenomenon before the entire gear start to loosen would of course be a benefit.

Adding new sensor technologies to the HUMS will require a profound redesign of the airborne segment. These are thus considerations which should be kept in mind for the long term system evolution.

Indicator Processing and Classification

The current classification methods evaluate the input of each indicator against an individual threshold. There is no evaluation of indicator trends over time, and no testing for parallel drift in indicators.

Most faults tend to cause gradual increase in indicator values, creating trends of a more or less clear nature. Further, most faults tend to affect more than one indicator, causing parallel trends on several indicators. For some faults, the indicators tend to rise also on the adjacent components. These are correlations that could be exploited to improve diagnosis results.

Given N indicators calculated at M acquisitions, the most general way of evaluating this information is an N by M feature matrix containing all information ever recorded. A single instance of this matrix will contain all information necessary to detect faults on all components on the aircraft. For simplification purposes, this operation can be split up in several steps, each step covering one component. N will then be replaced by a subset of N, N', containing all indicators for the component in question. N' might also contain indicators from adjacent components. M can be replaced by a subset of M, M', containing the last few acquisitions or all acquisitions since last overhaul.

Evaluation of these feature matrices can be performed by classification systems such as clustering, artificial neural networks, fuzzy logic, or polynomial approximation. As any component state will not have an unambiguous signature in such a matrix, it is necessary to perform a second level parameterization. This can be achieved by for instance calculating the indicator derivative or developing a parametric indicator progression model.

HUMS support personnel are capable of detecting faults more precisely

than any HUMS, simply by using manual screening of the condition indicators. This screening is mainly focused on indicator progression analysis. Automating this process will thus provide a substantial improvement of the HUMS, especially of it involves avoiding aircraft individual thresholds. As indicator progression analysis can be developed without any modification to the airborne segment, this axis of research should have the highest priority.

Data Migration

All first generation HUMS were made under the assumption that a given aircraft would be associated with a single ground station. Practice has however shown that HUMS data from a single aircraft can be processed at several ground stations, on the various bases of the operator. This creates obvious data consistency problems, as data from a single aircraft will be fragmented across several ground stations.

Another problem is moving data from the operator to Eurocopter. This is performed through backup tapes sent in the paper mail at more or less regular intervals. The procedure is however too slow to perform the customer has a potential problem. In response to possible detections, information is sent to Eurocopter by emailing indicator plot screenshots. This is cumbersome for the operator, and does not always provide Eurocopter support personnel with all the necessary information.

Developing a model which allows migration of HUMS data between ground stations and between ground stations and Eurocopter should be given high priority. Such a model is vital both to answer the clients day to day data migration needs, as well as to provide Eurocopter with a situation awareness concerning its HUMS equipped fleet. The latter point is vital to any further development the MARMS / EuroARMS systems, as it helps providing relevant data for research into fault detection algorithms.

3.7 Axis of Research

Based on the improvement potential identified in the previous section, this work follows several axis of research. The suggested improvements are however too numerous to be explored in the context of a single PhD. A decision was therefore made to focus on methods not requiring redesign of the airborne system. This leaves research into methods for improved processing and interpretation of the condition indicators. Further, a set of measures are proposed to deal with some of the data migration issues.

To reduce data scatter, contextual data correction has been developed

both for signals and for indicators. Contextual correction of indicators is aimed at indicators originating from short duration acquisitions, for which contextual variation within the acquisition is unlikely. Contextual correction of signals is aimed at long duration signals, and permits correcting a signal piece vice to compensate for context change throughout the period. The latter method remains theoretic as it can not be implemented and tested on the current generation airborne segment. It was developed none the less, due to its relevance for future use.

To avoid the current problems of individual indicator thresholds for each aircraft, two methods for indicator trend analysis were developed. These methods start by calculating traditional indicators from a batch of vibration acquisitions. Once an indicator series is obtained, trend analysis is used for analyzing how the indicator series behaves over time. The first method uses a flexible parametric model to approximate indicator behavior over time. Indicator behavior along the time line is then identified by evaluating the first derivative of this model. The second method applies a set of wavelet filter banks to the indicators separating regions representing maintenance actions, normality and mechanical degradation. The wavelet coefficients are then passed through a threshold system or a radial basis network to flag regions of maintenance actions and mechanical degradation.

Finally, a framework for HUMS data migration is developed. This framework is designed to facilitate transport of data between the operator and the HUMS OEM, and to help the HUMS OEM fuse together data recorded by different systems. The data migration framework is completed with a system for online registration of HUMS alarms and mechanical problems. This facilitates the communication between the operator and the HUMS OEM customer support, and greatly improves the response time for customer support as well as reducing operator workload.

Eurocopter France owns a patent, currently pending, on the work concerning contextual normalization and non-parametric trend analysis.

Chapter 4

Data Migration

4.1 Introduction

For the continued evolution of the Health and Usage Monitoring Systems, it is of vital importance to be able to aggregate the experience already obtained by the systems currently in service. For an airframe OEM, this involves collecting data at regular intervals from all of its fleet, typically involving multiple HUMS models and versions produced by delivered HUMS OEMs. Collecting and fusing data from different HUMS models and versions poses several technical challenges, as each system organizes its data storage in different ways. This chapter presents a data handling system which has been developed as part of this PhD study. The data handling system is designed to fuse the data from different systems and system versions into a common database, so that this data can be accessed through a single interface. Such a tool is essential for extracting and aggregating the experience obtained through all HUMS solutions currently and formerly in use [49].

4.2 Analysis Process

Although each HUMS solution on the market does things slightly different, the overall process is more or less the same (Fig. 4.1). This includes the datatypes that are recorded or derived by the HUMS. The fundamental data sources for usage monitoring are the flight data parameters, i.e. information like airspeed, altitude and engine torque. These are used for generating event markers like engine overheat and rotor overspeed. The usage monitoring function also calculates usage cycles, and generates event markers for components reaching their retirement age in terms of accumulated cycles. Each marker is stored with the contextual information relevant to its type.

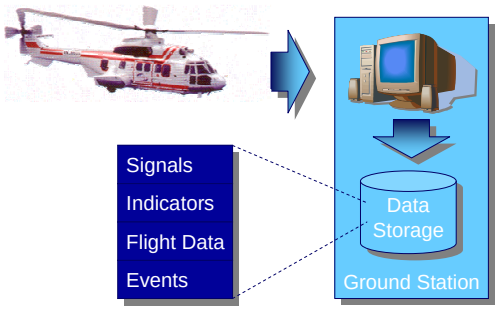

Figure 4.1: HUMS analysis process.

For health monitoring, the fundamental data sources are the vibration recordings. From the recording from each component, a set of condition indicators are derived. These are parameterizations of the raw vibration signals, and are closely correlated with the state of the underlying assets. The indicators are aggregated by a classification system able to detect the presence of mechanical degradation. A set of event markers are produced by the classification system. This set of markers represents observations of assumed mechanical degradation, and compliments the set of markers representing anticipated faults generated by the usage monitoring.

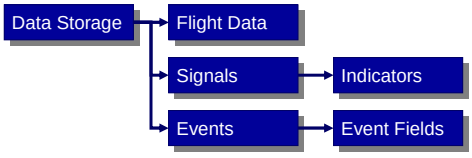

Figure 4.2: Basic datatypes.

This makes a total of five datatypes for the HUMS. Two fundamental once; flight data parameters and vibration recordings, and three derived once; indicators, event markers and event marker fields (Fig. 4.2). The event marker fields are the contextual information companying each event marker. The work distribution between the airborne segment and the ground station is proprietary to each HUMS solution. Regardless, all five datatypes will eventually end up in the data storage of the ground station.

4.3 Architectural Layers

Although the data from every HUMS can be generalized into a set of standard datatypes, the storage format in proprietary to each HUMS. Most HUMS solutions on the market today use either a proprietary directory / file structure or a third party SQL database (Sec. B.1). Even though SQL databases have standard interfaces, the table structure is still proprietary to each HUMS version.

This barrier has been overcome by developing a data storage driver for each supported HUMS version. A storage driver is an implementation of standardized Application Program Interface (API). The functions defined in the API provide the necessary tools to connect to a data storage, enumerate the elements of each datatype, and extract the underlying data. Although the inner workings of each driver is very different, the outside interfaces are identical (Sec. B.2.1).

The common interface layer exposes this standardized interface to any third party application through ActiveX (Sec. B.2.3) and .net (Sec. B.2.4). This layer accepts connection requests from any third party application, loads the driver corresponding to the HUMS version provided in the connection request, and connects to the specified target data source. Once a connection is open, the third party application can interrogate data objects within any supported HUMS data storage without any knowledge about its underlying structure (Fig. 4.3). The choice of Microsoft-based component models, over more open solutions like Java (Sec. B.2.2), was made due to the fact that large amounts of code has already been developed inside the company base on Microsoft computing platforms.

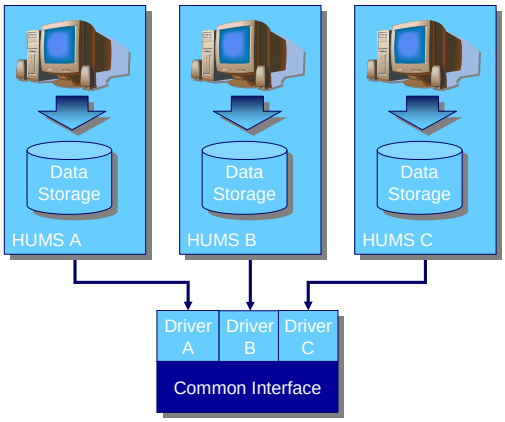

Figure 4.3: Data interface layers.

All HUMS data drivers support remote connection, meaning that data source and target application need not be running on the same computer. For HUMS solutions relying on a third party SQL database, remote access is managed by the database engine and connectivity drivers. For proprietary directory / file structured data storage, remote access is provided through standard file sharing protocols. This way, remote access is provided without having to install any additional software on the data storage servers. This is an important point, as ground station software today exists for a variety of hardware and operating system platforms which in some cases are in the process of becoming obsolete, thus rendering development of any additional platform specific software unpractical.

For communication across Internet, data are channeled through Virtual Private Networks (VPN). VPN is an open technology which ties together two Internet connected Local Area Networks (LAN) so that they appear as one. A VPN tunnel ensures protection of both end networks, provides authentication of both parties, and allows encryption of all data transferred through the tunnel. Again, this is achieved using only standard protocols which are supported by all platforms.

4.4 Common Storage

The continuous research into fault detection algorithms requires large amounts of vibration data in order to understand how mechanical faults affect the vibration signatures of the transmission system. Test-rig experiments are expensive, and are not necessarily representative for the vibration signatures recorded in flight. It is thus essential for a HUMS OEM to gather as much authentic HUMS data as possible. Such data can be extracted from the operator ground stations using the solution presented in the previous section.

An additional technical challenge is finding efficient means of storing large quantities of data from several systems. By default, a HUMS OEM must keep one data reservoir available for each system and version in order to provide its researchers with access to the relevant data. Further, some ground station data reservoirs have limited capacity due to their design, meaning that a large number of data storage servers are necessary to host the data from an entire fleet of aircraft.

This problem has been overcome by the development of a common data storage. The common data storage is based on the generalized HUMS data format introduced in the previous sections, and is thus capable of storing data from any system for which a data reservoir driver exist. Further, this data storage solution is scaled to store all data ever recorded by an entire

fleet. Consequently, it provides researchers with a single interface into all data recorded by all supported systems, greatly simplifying data management tasks for the HUMS OEM.

A data synchronization tool allows data to be transferred from a source ground station to the common data storage. The synchronization tool has its own graphical user interface (GUI) for selecting source and target connection credentials. In addition, it has an ActiveX programmatic interface allowing integration into other computer systems, such as web fronts.

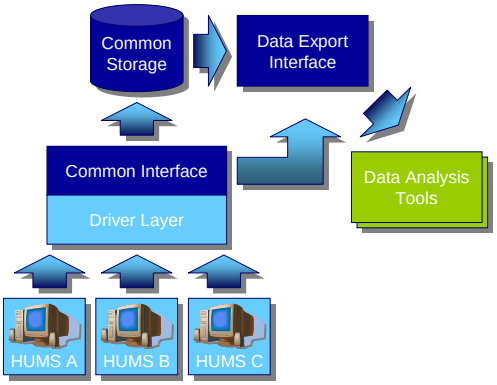

Figure 4.4: Global dataflow.

Rather than interfacing directly with the common data storage at SQL level, third party applications gains access through a data export interface. The data export interface is an ActiveX component allowing third party application to programmatically enumerate the objects within the common storage and extract the underlying data (Fig. 4.4). It has also a GUI allowing the extraction of data into applications which do not have their own GUI for data object management. Further, the data export interface offers a mechanism for connecting directly to a ground station data reservoir through the common interface layer. This is especially convenient in support situations, as it allows HUMS OEM support personnel to tap directly into an operator's ground station using specialist data analysis tools.

4.5 Discrepancy Reporting

Historical HUMS data is of little use if the states of the assets corresponding to the data recordings are not known. In order to extract any knowledge about the correlation between mechanical states and vibration signatures, both the states and their corresponding signatures must be known. In the

context of pattern recognition, this is known as marked training sets. In order to have the historical HUMS data from an aircraft correctly marked, it is necessary to know when the operator experienced mechanical problems, as well as the nature of the problems.

This has led to the development of a discrepancy reporting system. The discrepancy reporting system allows operators to file a report online when ever an anomaly is suspected. Any such report will be handled by an expert at the HUMS OEM advising the operator of the appropriate action. This is a two-way communication process where the operator and the HUMS expert work together to locate the origin of the problem (Fig. 4.5). Once the problem is identified and the appropriate actions taken, the HUMS expert will set a marker in the common storage database explaining the uncovered anomaly, if any. This marker will be stored for future reference with all communication made during the fault isolation process.

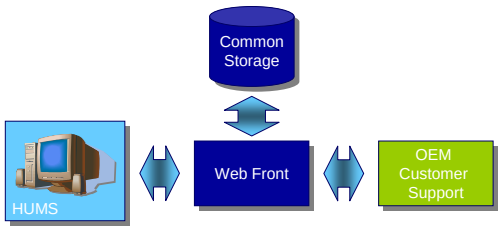

Figure 4.5: Discrepancy reporting and follow-up.

Discrepancy reporting procedures already exists in some form or another for most HUMS OEMs. The advantage of the online reporting system is that it cuts response time for the support personnel significantly. Further, it provides automatic logging of discrepancy reports, facilitating statistical studies and correlation with HUMS data. This combination of a HUMS data storage facility and a discrepancy database allows researchers and support personnel to extract and investigate the vibration signatures corresponding to specific mechanical problems through the click of a button. When managing data from an entire fleet of aircraft, each aircraft producing hundreds of condition indicators for thousands of hours every year, it is essential to have data management at this level in order to keep oversight.

4.6 Benefits

This data storage facility is meant to benefit both researchers and support personnel. The key advantage for support personnel is the speed and flexibil-

ity of the discrepancy reporting system, allowing faster response to operator requests. This is supplemented by the common interface which gives instant access operators' HUMS data. Instant access means that support personnel can view an operator's HUMS data directly, rather than explicitly requesting the operator to send the necessary data (Fig. 4.6). This results in reduced workload for the operator and shorter response time for support personnel. An additional benefit is that the periodic data transfer from operator to the HUMS OEM no longer requires the intervention of the operator, again reducing operator workload.

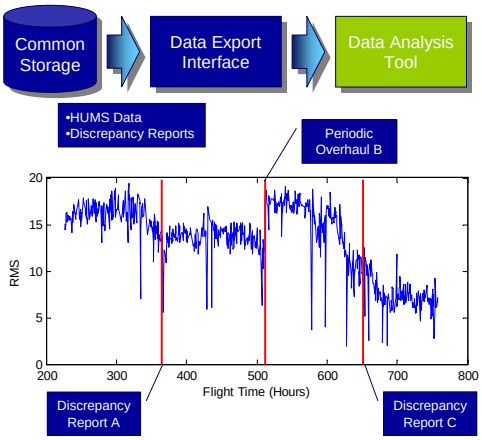

Figure 4.6: Condition indicator with discrepancy markers.

The motivation for providing researchers with this tool is, as already mentioned, to facilitate research into the correlation between vibration signatures and drive-train failure modes. Further, the discrepancy database permits data mining operations like calculating fault detection frequencies and false alarm frequencies. Such frequencies can be calculated across indicators, components and rotorcraft models. This helps the HUMS OEM gain a better understanding of the effectiveness of each HUMS function, so that research can be focused on the most significant weak-points. It also gives support services a better situation awareness concerning problem distribution across operators, aircraft and aircraft models.

4.7 Conclusion

Most in-service difficulties associated with Health and Usage Monitoring Systems can be attributed to their complexity. The number of components monitored and the number of failure modes these systems are designed to detect

are immense. This makes it highly challenging to monitor the performance and reliability of each sub-function, and poses the risk of HUMS OEMs being buried in discrepancy reports and raw HUMS data without being able to extract the knowledge contained within.

The data migration solutions developed in this thesis are an attempt to counter these problems by aiding the HUMS OEM in organizing the incoming data, and extracting its essence. This is done by addressing the problem of accumulating data extracted from different systems, correlating this data with discrepancy logs, and exporting it to third party numerical analysis tools. Although the two latter points are already addressed by certain OEMs in some form or another, data fusion across systems is an area which so far has received little attention. This is however a vital point, given the number of solutions in service today. For the continued evolutions of HUMS, it is essential to be able to exploit the knowledge accumulated by older systems when developing the next generation technology.

The tradeoffs from better data handling are both short and long term. On a short horizon, these methods provide better support services for the HUMS OEM by reducing response time for the support personnel as well as reducing operator workload. In the long run, better data handling will result in better situation awareness for the HUMS OEM, making it easier to adapt the systems to customer demand.

Chapter 5

Data Correction

5.1 Introduction

The spectral signature of HUMS vibration acquisitions are affected not only by the underlying assets, but also environmental factors [50]. During operation, acquisitions are performed at different airspeeds, engine torques and oil-temperatures, as well as during level flight, turning, climbing and so on. As the environmental context of an acquisition is random, relative to when the acquisition is performed, the impact of the various conditions is manifested as random variations between the signals. This impact is manifested differently for each frequency on each acquisition. The energy at some indicators / frequencies at some acquisitions are heavily influenced by environmental factors, while others are not.

These random variations are manifested as noise clouding the vibration measurements. Working with fault detection, it is desirable to reduce random noise as much as possible, in order to avoid erroneous diagnosis as a result of unreliable measurements. Methods to limit contextual variations in the measurements currently implemented in commercial solutions are mainly limited to contextual windows for when acquisition is enabled. I.e. the use of min and max criteria for signature-influential parameters, such as airspeed and torque. A disadvantage of this approach is that variations can still be considerable within these windows. Further, applying strict contextual windows poses problems for aircraft with diverse operating envelopes, such as search and rescue aircraft, resulting in low flight time within the contextual window and consequently a low data volume per flight.

This chapter tries to compensate for contextual variations through modeling. After an initial theoretical framework is developed, methods for both indicator correction and raw signal correction are presented. The purpose

of the methods is to de-correlate the vibration signatures and their environmental context, thus reducing variance between observations representing the same condition of the underlying asset. The better choice of correction method, indicator correction or raw signal correction, depends on the type of indicator and diagnosis methods are deployed on the corrected data, and will be discussed in the following.

5.2 Modeling

In this study, it is assumed that the observed finite length signal x recorded at time t can be seen as the product of a number of models M_k, each depending on the linear or nonlinear combinations of the elements in a vector of model parameters $p_k(t)$ (Eq. 5.1).

$$x(t) = \prod_k M_k(p_k(t)) \tag{5.1}$$

It is further assumed that this expression can be simplified by considering only the influence by the condition of the associated component M_c and the environmental factors M_e (Eq. 5.2).

$$x(t) = M_c.M_e(p_c(t)) \tag{5.2}$$

The environmental influence, M_e, is given by the environment at the time of acquisition. Each HUMS acquisition is accompanied by a set of contextual parameters describing this environment. These are flight data parameters such as airspeed, torque and oil temperature, where the selection of parameters available depends on HUMS model and version. Consequently, M_e can be made as a function of an array of contextual parameters $p_c(t)$.

Defining x_t as a set of finite-length signals, makes x a vector of signal samples and t acquisition start time. With x_t acquired from a component in a stationary condition throughout the set, the output from M_c will be constant across t. Consequently, the only factor contributing to non-constant behavior in x_t across t is M_e. Given x_t and the contextual parameters on which M_e depends, the function M_e can be estimated. This provided that all the necessary contextual parameters are recorded, and that sufficient relevant training data exists.

In order to cancel the effect of environmental changes, a reference environment must be defined. The purpose of the reference environment, a vector of contextual parameters constituting an environment of reference p_e^r, is to correct each observation so that they appear to have been made in this environment. A correction function (Eq. 5.3) is defined so that it for

an observation provides the ratio between the reference environment and the environment at the time of the observation. Weighting each observation with its correction function G will thus de-correlate M_e and the observations.

$$G(p_e(t)) = \frac{M_e(p_e^r)}{M_e(p_e(t))} \tag{5.3}$$

This methodology can be applied directly to a signal, or to each indicator derived from the signal of a given component. The former approach requires G to be a filter, where the filter transfer function is given by the set of relevant flight parameters. Using the latter method makes G a simple scalar function describing the coupling between a single indicator and the set of relevant flight parameters. A function G must thus be estimated for each indicator of the signal associated with each component on the aircraft.

This chapter makes no assumptions about the underlying physical phenomena responsible for the correlation between environmental context and vibrations signature. The methods developed here are purely general, and must be adapted to each component on the aircraft.

5.3 Indicator Correction

Not all indicators are sensitive to environmental changes. Others are sensitive, but show a change in scatter rather than localization. A significant group of indicators show a substantial change in location as a function of environmental context. The relationship between indicators from this group and the applicable flight parameters is normally possible to approximate with a polynomial model (Eq. 5.4). In this case $p_e(t)$ contains not only the parameter, or parameters, of interest, but also the necessary powers for each parameter.

$$\hat{M}_e(p_e(t)) = p_e(t).\hat{a} \tag{5.4}$$

The vector $\hat{a}$ contains the weight of each power of each parameter, and is estimated using a set of indicator values i and their associated flight parameters p_e recorded over a period where the condition of the underlying asset is stationary. As the condition is stationary, any fluctuations in the indicator value must be caused by environmental variations. By subtracting the mean value of the indicator μ_i, the fluctuations are isolated, and the model $\hat{M}_e(p_e(t))$ is estimated to approximate these fluctuations (Eq. 5.5). By subtracting the environmental model from the indicator series, the fluctuations caused by environmental changes are removed, thus reducing indicator scatter.

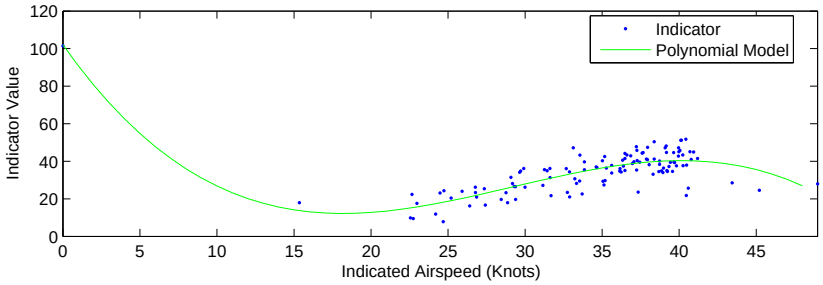

Figure 5.1: LH Free Wheel Gear RMS versus indicate airspeed.

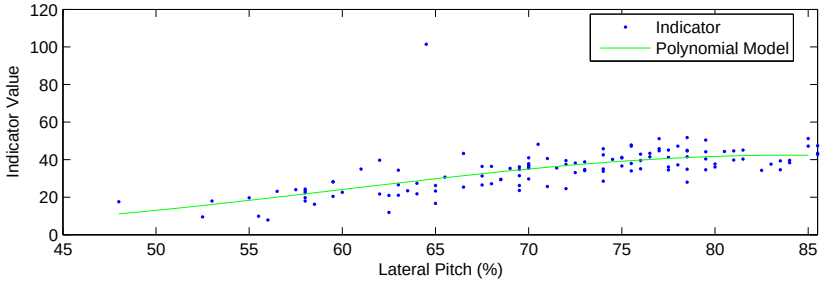

Figure 5.2: LH Free Wheel Gear RMS versus lateral pitch.

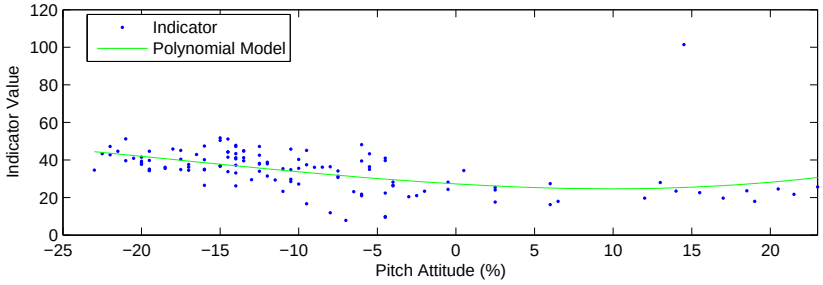

Figure 5.3: LH Free Wheel Gear RMS versus pitch attitude.

$$\hat{a} = (p_e p_e^T)^{-1} p_e^T . (i - \mu_i) \tag{5.5}$$

Figures 5.1, 5.2 and 5.3 shows the left hand free wheel gear rms indicator from an EC225 in slow cruise as a function of indicated airspeed, lateral pitch, and pitch attitude. Indicated airspeed is the speed of the aircraft relative to the atmosphere, lateral pitch is the cyclic stick position in the lateral (forward) direction, and pitch attitude is the angle of the aircraft in the forward direction. The unit is knots for the first parameter, and percentage of max angle for the two others. During cruise, these three parameters are strongly correlated. The data for this example was however acquired in slow cruise. In such condition there is a substantial delay between a change in stick position, subsequent change in aircraft angle, and finally change in aircraft speed. The three parameters are thus only partially correlated for this dataset. Common for all three parameter is however their correlation with engine torque.

From the figures, it appears as if there is a strong correlation between all three parameters and the indicator. The green line in each figure shows a third order polynomial approximation of the relationship between indicator and parameter.

Figure 5.4 contains the above indicator as a function of acquisition index, with the raw indicator accompanied by a corrected one. Model estimation was done using acquisitions 50 to 110. As can be seen from the figure, the model remains valid also outside the training period.

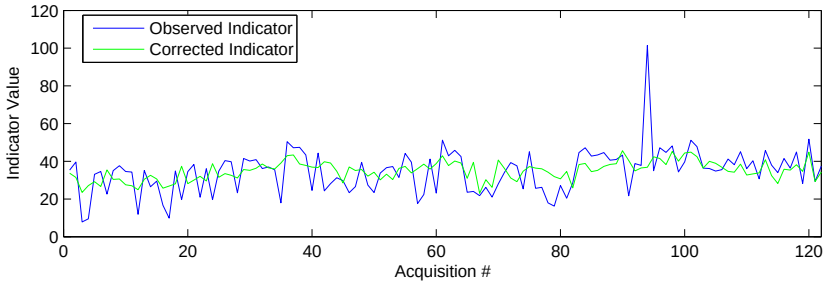

Figure 5.4: LH Free Wheel Gear RMS raw and corrected for environmental changes.

The relationship between indicator values and environmental conditions is specific to each indicator and aircraft. In the above example, there is a certain correlation between the three parameters. All three parameters are also known to have an impact on torque, which is probably the underlying cause of the indicator fluctuations. Physical models for rotorcraft dynamics

and their impact on vibration signatures are however outside the scope of this study.

5.4 Signal Correction

Working directly on the signals, it is only necessary to estimate one model per component, although this argument is countered by the increased complexity of this method. An advantage held by direct signal correction is that it is producing corrected raw signals suitable for non-linear indicators and classification methods working directly in the time or frequency domains [46]. Examples of such are adaptive lifting [41] and mathematical modeling [14]. These methods use a reference wavelet or filter as an approximate of the signal, calculating the distance between each sample signal and the reference. Once a scalar feature is extracted, like the sum square difference, it is too late to perform any correction.

Another advantage is the possibility to piecewise normalize the raw signals before any further processing is performed. This is of interest for acquisitions where the recording period is sufficiently long for the environmental context to be subject to change throughout the recording period. Components requiring acquisitions of long duration are mainly rotors, as these rotate at slow speed.

This section attempts to de-correlates environmental context and signal power spectrum magnitude, giving the impression that all signals where acquired in the reference environment. Any correlation between environmental factors and power spectrum phase is however not considered. On the contrary, the correction filter does itself introduce a significant phase distortion to the signal. If this is acceptable or not, depends on the classification system for which the data is intended. Most systems in use today rely only on signal magnitude at specific frequencies, and does not consider phase. Should the above method be used for pre-processing data for a phase-sensitive classification system, the data must also be passed through a phase-equalizer correcting the distortions caused by the magnitude-equalizer.

To de-correlate vibration power spectrum and environmental context, it is necessary to create a model describing the environmental impact on the signal waveform. This can be done by evaluating the signal Power Spectral Density (PSD) as a function of significant environmental factors, for example airspeed as shown in figure 5.5. Note that frequency is given in shaft order. For this data set, a non-parametric PSD is obtained using a simple discrete Fourier transform, as the signals have already been averaged in the time-domain. The signal PSD magnitude and phase is thus an alternative representation

of the time-domain signal, without any loss of information.

To model the spectral behavior as a function of the environmental context, it is however necessary to approximate the signal PSD using a parametric model. As seen in figure 5.5, gear vibration signals consist of a small number of high-energy regions, in this example only one, corresponding to the gear meshing harmonics and modulation sidebands, over a noise-floor. Such a spectral shape can successfully be approximated by an autoregressive (AR) model (Eq. 5.6). An AR model has a number K of high-energy regions, poles, over a base floor. The frequency position of each pole is given by $\omega_k \in [0, 2\pi]$, while the energy level is controlled by $r_k \in [0, 1\rangle$. The general level is given by b_0. All complex poles ($\omega_k \notin \{0, pi\}$), must have a complex conjugate, or output will be complex.

$$H(\omega) = \frac{b_0}{1 + \prod_{k \in K} r_k e^{j\omega_k} e^{-j\omega}} \tag{5.6}$$

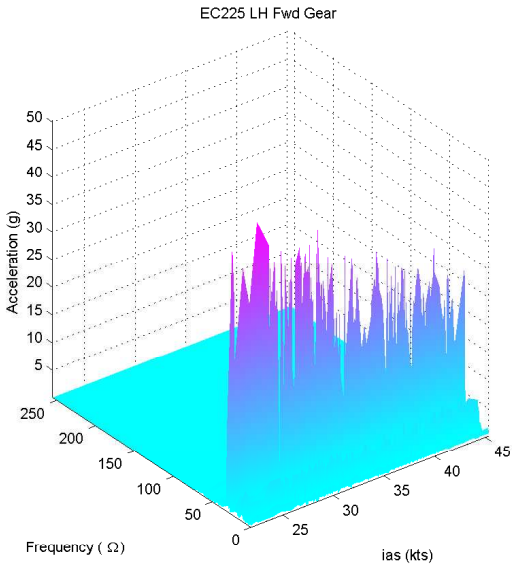

Figure 5.5: Magnitude PSD of fwd gear acquisitions order by speed.

The amplitude of the high-energy regions vary with the environmental context while the positions in frequency is constant. Consequently, it is possible to use a simplified model which explicitly defines ω_k for each component and optimizes only b_0 and r_k. An altered version (Eq. 5.7) of the original AR prototype is defined, forcing every pole to have a complex conjugate. This simplification can be made without loss of generality as no acquisitions have meshing tone harmonics or modulation sidebands at dc or π frequency,

meaning that all poles representing meshing tone harmonics or sidebands must have a complex conjugate. The adjustable parameters b_0 and r_k are estimated using Trust Region (Sec. A.4.1). This estimation can also be performed using an evolutionary algorithm or other gradient-based methods.

$$H^{(p_e)}(\omega) = \frac{b_0}{1 + \prod_{k \in K} z_k^{(p_e)} e^{-j\omega} z_k^{*(p_e)} e^{-j\omega}} \tag{5.7}$$

$$z_k^{(x)} = r_k^{(x)} e^{j\omega_k^{(x)}} \tag{5.8}$$

As an alternative, it is still possible to use the original AR definition and a textbook estimator like LPC or Burg [36]. This will however require an algorithm for keeping track of the pole angles relative to their indexes, as these might change order from signal to signal using a textbook optimizer.

The number of complex conjugate poles is chosen to match the number of high-energy regions, and the pole angles ω_k are set to match the frequency of these regions. By estimating each signal $X^{(p_e)}(\omega)$ in the dataset, the corresponding approximate $H^{(p_e)}(\omega)$, given by $b_0^{(p_e)}$ and $r_k^{(p_e)}$, are obtained (Eq. 5.9). The variable $E^{(p_e)}(\omega)$ is approximation error.

$$H^{(p_e)}(\omega) = X^{(p_e)}(\omega) - E^{(p_e)}(\omega) \tag{5.9}$$

Figure 5.6 shows the magnitude PSD of the same set of signals as in figure 5.5, but with each signal $X^{(p_e)}(\omega)$ replaced by its AR approximate $H^{(p_e)}(\omega)$. The parameters making up each AR model, $b_0^{(p_e)}$ and $r_k^{(p_e)}$, can themselves be modeled as a function of the contextual parameters p_e using a parametric model. This example used a third order polynomial model, although this might not necessarily be the optimal choice for the pole radius, as this parameter is always between zero and one. A better model for this parameter might be a sigmoid, or some other function with output confined between zero and one.

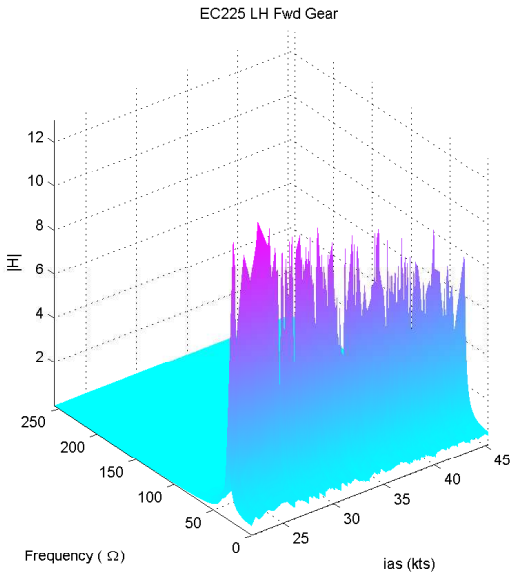

Figure 5.6: Fwd gear $H^{(p_e)}(\omega)$.

The meshing tone amplitude is significantly smaller for $H^{(p_e)}(\omega)$ than the non-parametric PSD. This is because $b_0^{(p_e)}$ and $r_k^{(p_e)}$ are estimated in the least-square-error sense, and the meshing tone represents only a single point in the PSD. If need be, this problem can be amended by giving the meshing tone frequency higher weight than the rest when optimizing, though at the cost of less precision for the other frequencies. The difference in amplitude is however of less importance, as it is proportional for all values of p_e and it is the ratio between different values of p_e that is of interest.

$$\hat{H}^{(p_e)}(\omega) = \frac{\hat{b}_0^{(p_e)}}{1 + \prod_{k \in K} \hat{z}_k^{(p_e)} e^{-j\omega} \hat{z}_k^{*(p_e)} e^{-j\omega}} \tag{5.10}$$

Replacing the filter parameters $b_0^{(p_e)}$, $r_k^{(p_e)}$ and $\omega_k^{(p_e)}$ by their polynomial approximates, $\hat{b}_0^{(p_e)}$, $\hat{r}_k^{(p_e)}$ and $\hat{\omega}_k^{(p_e)}$, the model $\hat{H}^{(p_e)}(\omega)$ is obtained, modeling signal energy both as a function of frequency and airspeed (Fig. 5.7) (Eq. 5.10).

In order to correct the signals, a reference environment p_e^r is chosen. The correction filter $G^{(p_e)}(\omega)$ represents the ratio between the reference power spectrum and the power spectrum corresponding to any contextual environment. Due to the division of two AR filters, the correction filter (Eq. 5.11) becomes an autoregressive moving average (ARMA).

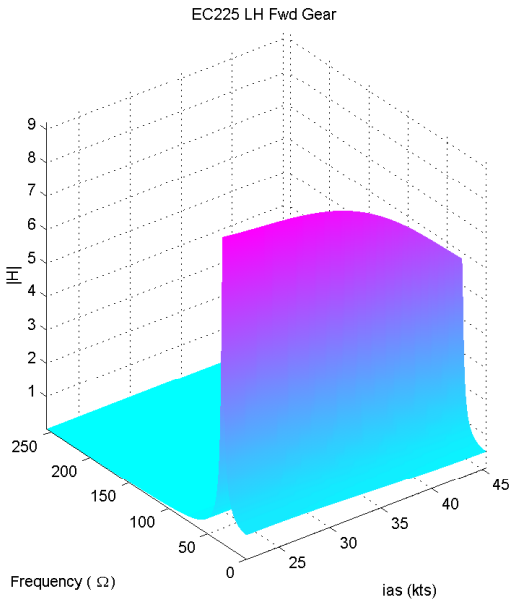

Figure 5.7: Fwd gear $\hat{H}^{(ias)}(\omega)$.

$$G^{(p_e)}(\omega) = \frac{\hat{H}^{(p_e^r)}(\omega)}{\hat{H}^{(p_e)}(\omega)} \tag{5.11}$$

$$G^{(p_e)}(\omega) = \frac{\hat{b}_0^{p_e^r}(1 + \prod_{k \in K} \hat{z}_k^{(p_e)} e^{-j\omega} \hat{z}_k^{*(p_e)} e^{-j\omega})}{\hat{b}_0^{(p_e)}(1 + \prod_k \hat{z}_k^{(p_e^r)} e^{-j\omega} \hat{z}_k^{*(p_e^r)} e^{-j\omega})} \tag{5.12}$$

The surface of the correction model (Eq. 5.12) is so that if multiplied with the airspeed model (Eq. 5.10), it returns the reference power spectrum for all values of p_e. The time-domain filter coefficients for a given p_e, b_k and a_k, is the coefficients corresponding to the numerator and denominator polynomials of $G^{(p_e)}(\omega)$. Deriving a correction filter from the airspeed of each signal, and applying them before further processing is performed, will largely remove the signal's contextual sensitivity (Fig. 5.8).

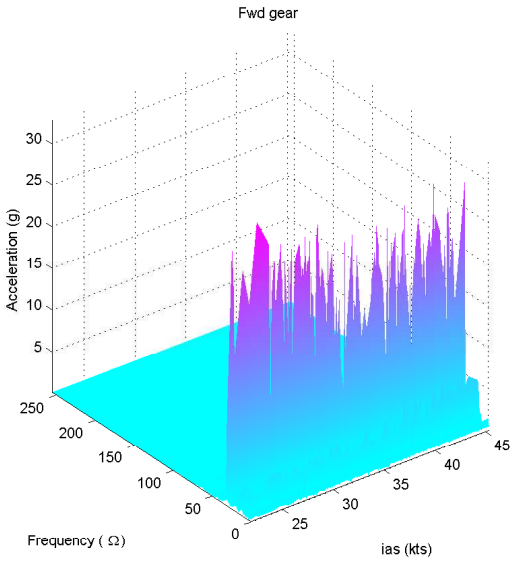

Figure 5.8: PSD of corrected fwd gear acquisitions order by speed.

5.5 Conclusion

The methods introduced here are an attempt to correct the environmental influence on HUMS vibration data, so that data acquired at different conditions are more comparable. This is favorable for all rotorcraft, especially those spending much time outside cruise, as it permits reducing data scatter and increase the overall reliability of the system. Also aircraft operating in near optimal condition can benefit from having its data corrected for environmental changes.

The underlying physical explanations for these correlations have not been addressed during this study. A subject for further research could be to establish a theoretical framework explaining why these correlations occur, and perhaps correlate indicators' environmental dependency to factors like incorrect equipment installation.

Chapter 6

Feature Extraction

6.1 Introduction

Most HUMS in use today evaluate each vibration measurement independently, with minimal correlation of these measurements along the time line. Progression analysis taking place in commercial HUMS is mainly limited to requiring N out of M indicator values to breach their threshold, this to avoid spikes setting off false alarms. This is in sharp contrast to the methods used by human HUMS analysts, which for most indicator types pay more attention to the progression of the indicator than its absolute value.

This chapter looks into progression analysis of vibration data as a means of feature extraction. A theoretical framework for the relationship between asset states and vibration signature progression is developed, as well as methods to analyze the progression of an observed indicator. Both parametric and non-parametric approaches to progression analysis has been explored, with strengths and weaknesses discussed in the chapter conclusion.

6.2 Progression Analysis

As explained in chapter 3.3.1, the gear vibration signatures are given by the matrices A and B in equations 3.2, 3.3 and 3.4 [30]. From this understanding of vibration signatures, gear condition estimation is transformed into a system identification problem. Each condition a given gear can exhibit has its set of values for A and B. Thus, by estimating these matrices, it is possible to uncover the corresponding condition. Although this equation set is underdetermined, making an unambiguous estimation impossible, it is fairly straight forward to specify a set of condition indicators capturing the essence of A and B. Consequently, the condition of a gear is given by its set of relevant

indicators. Also for bearings and shafts, a relatively small set of indicators is sufficient to discriminate all conditions these components can inhibit.

Vibration based fault detection for mechanical components is typically based on estimating the normal state vibration signature i.e. the normal state values for a set of relevant indicators, and comparing subsequent observations to this baseline. An observation displaying significant deviation from the normal state baseline must be considered as an observation of a component in an abnormal condition. The challenge is estimating the baseline, i.e. the normal state envelope for the set of relevant indicators. This because the vibration signature of a mechanical component is specific not only to each design, but specific to each physical realization of a given design. The cause of this stems from microscopic differences in the way each component is forged and mounted. All component types suffer from this problem, although gears typically have larger variation in vibration signature between individual realizations than bearings and shafts. In any case, the normal state baseline must be estimated for each component, and re-estimated after major overhauls.

As the condition of a component degrades, its condition changes as a function of time. Consequently, its vibrations signature and its set of relevant indicators changes as a function of time, where the function is determined by the failure mode. Observing a segment of a set of indicators, it is possible to estimate the function, or progression pattern, to which the segment corresponds. The progression pattern estimate will in turn provide a pointer to which conditions the observed component is traversing, and to which failure mode this set of conditions corresponds. These assumptions form the base for indicator progression analysis as a feature extraction tool.

6.2.1 Basic Progression Types

From a HUMS analyst's point of view, the progression of an indicator belongs to one of three classes; normal, step or trend. Indicators in the normal class have a constant expected value, although some indicators have considerable scatter on around of this mean. A step is a sharp transition between two levels, and is usually caused by maintenance actions. This corresponds to a reset of the indicator / condition model, as the set of indicator values corresponding to a given condition changes. Consequently, when working on fault detection methods using the absolute indicator values directly, any model must be re-estimated to compensate for this. The trend class represents a gradual increase or decrease in the expected value, and is usually associated with mechanical degradation, i.e. a traversal through component conditions.

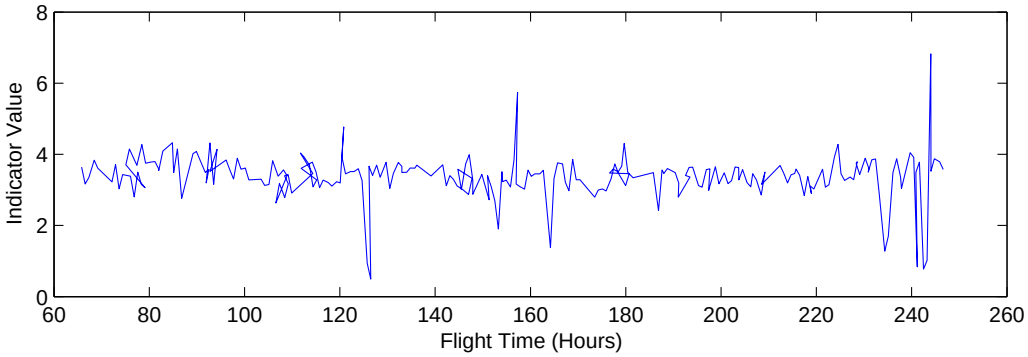

Figure 6.1: Normal indicator behavior.

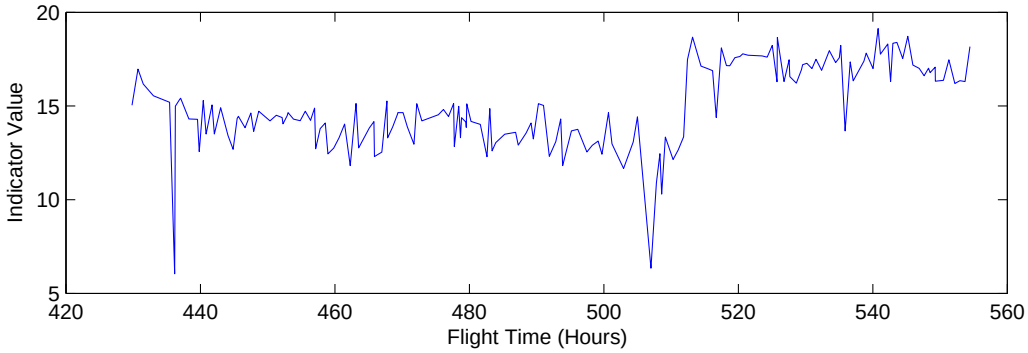

Figure 6.2: Step change due to maintenance.

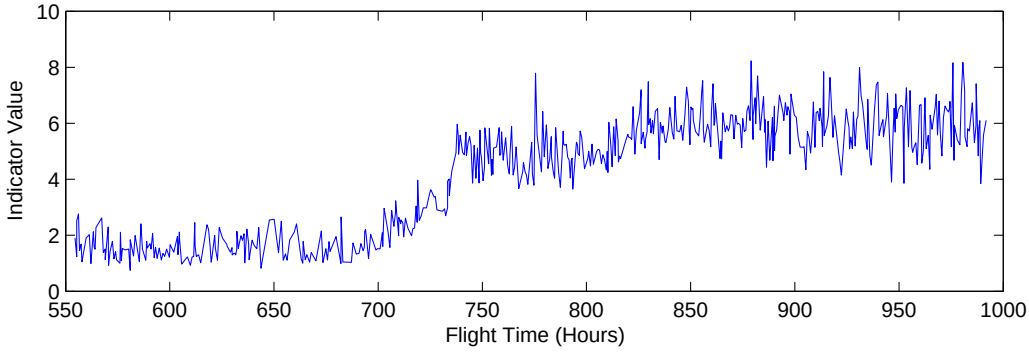

Figure 6.3: Trend due to component degradation.

Mechanical degradation is also frequently associated with an increase in indicator scatter. Figures 6.1, 6.2 and 6.3 contain examples and characteristics of normal, step and trend behavior.

Trying to model the progression of an indicator (Eq. 6.1), the time se-

ries can be split in a deterministic component d and a random process r. Although the contextual normalization introduced in the previous chapter removes some of the outliers and gaussian noise on the indicators, the data still remains noisy. This due to influence by processes which cannot be properly modeled and predicted. This scatter is thus labeled as the random process r. As established in the previous section, any change in component condition will cause changes in the value of d and / or in the gain of r, for one or more indicator associated with the component.

$$i(t) = d(t) + r(t) \tag{6.1}$$

Looking at figure 6.3, it is clear that the indicator series produced by this fault cases consist of several transitions, showing that the associated asset is traversing several conditions, or fault propagation stages. In an operational environment it is however impossible to observe the changes taking place in a mechanical system hour by hour. It is only the final result observed when a gearbox is removed and stripped which can be matched to its indicator progression pattern. As a given end result, or observable failure mode, can have several propagation patterns, it is difficult to match a known failure mode to an observed propagation pattern. Further, it is from an operational point of view sufficient to know if a component is in a healthy condition or not. If a component is suspected faulty, it will in any case be removed and inspected manually. Assuming that the initial condition of a component is healthy, it is for fault detection sufficient to detect any change in component condition, corresponding to changes in d and / or changes in the gain of r.

6.2.2 Progression Modeling

From the assumptions made in the previous section, an indicator progression model is constructed. For normal component behavior, the model produces a d with constant value and a r with constant gain. In response to a component replacement, the model produces an abrupt change in the value of d. To emulate progression patterns associated with mechanical degradation, the model produces random transitions in the value of d and in the gain of r.

The variations in d are obviously not random for a given failure mode. However, given a failure mode which is unknown or not well understood, these fluctuations will appear as random. As this is the case for most failure modes to which a helicopter transmission system is susceptible, the fault signature model is made without assumptions of a priori knowledge of progression patterns.

The two main components for the indicator time series is the deterministic

process d and the random process r (Eq. 6.1). The component d is itself made up of a step component b and a trend component c.

$$d(t) = b(t) + c(t) \tag{6.2}$$

The component b is recursive with initial value $b(0)$ equal to dc_c. This constitutes the initial value of b and thus the initial expected value of the indicator. The last term of the expression contains a boolean expression which returns 1 when true, and 0 otherwise. This results in a step of amplitude a_b at position p_b. The parameters a_b and p_b can be arrays of several elements, $a_b^{(k)}$ and $p_b^{(k)}$, in which case several steps will be generated and step index is denoted by k.

$$b(t) = b(t-1) + [t == p_b^{(k)}].a_b^{(k)} \tag{6.3}$$

The trend component is made up of a sum of weighted sigmoids, where a_c constitutes amplitude, q_c slope and p_c position in time. The variable k is the sigmoid index, and K_c is the number of sigmoids in the sum.

$$c(t) = \sum_{k \in K_c} \frac{a_c^{(k)}}{1 + e^{-q_c^{(k)}(t - p_c^{(k)})}} \tag{6.4}$$

The random component r is itself made up of two processes, the outlier component s and the white noise process w.

$$r(t) = s(t) + w(t) \tag{6.5}$$

The white noise component is constructed from a gaussian process r_g with variable gain g_w (Eq. 6.6). The gain function g_w is given by a sum of sigmoids over a constant (Eq. 6.7).

$$w(t) = g_w(t).r_g \tag{6.6}$$

$$g_w(t) = dc_w + \sum_{k \in K_w} \frac{a_w^{(k)}}{1 + e^{-q_w^{(k)}(t - p_w^{(k)})}} \tag{6.7}$$

The outlier component is constructed from a gaussian distribution r_g (Eq. 6.9) with gain g_s times the gain of w. Any point smaller than two standard deviations are then set to zero (Eq. 6.8), so that only the peak values are kept.

$$s(t) = s'(t).[s'(t) > 2g_s.g_w(t)] \tag{6.8}$$

$$s'(t) = r_g.g_s.g_w(t) \tag{6.9}$$

This gives a total of 13 configuration parameters controlling the characteristics of a synthetically generated indicator series: dc_c, $a_b^{(k)}$ and $t_b^{(k)}$ controlling the edge process, K_c, $a_c^{(k)}$, $q_c^{(k)}$ and $p_c^{(k)}$ controlling the trend process, dc_w, K_w, $a_w^{(k)}$, $q_w^{(k)}$ and $p_w^{(k)}$ controlling the white noise process, and g_s controlling the outlier process. By assigning specific values or probability distributions to these parameters, indicator progressions similar to those associated with various mechanical conditions can be generated.

Figure 6.4 shows a flow chart of the indicator model. The collection boxes (double boxes) used for sigmoids and steps signifies that zero or more transitions are generated. In the cases where more shapes a produced, these are simply added together in the sum blocks following each sigmoid and step block. When several shapes a produced by a collection block, the control parameters for each transition are tagged with transition index (k) in superscript, as shown in equation 6.3, 6.4 and 6.7. The operator "$>$" produces zero for false and one for true. The intermediate results g_w, w, s, b, c, d and r are labeled in the figure.

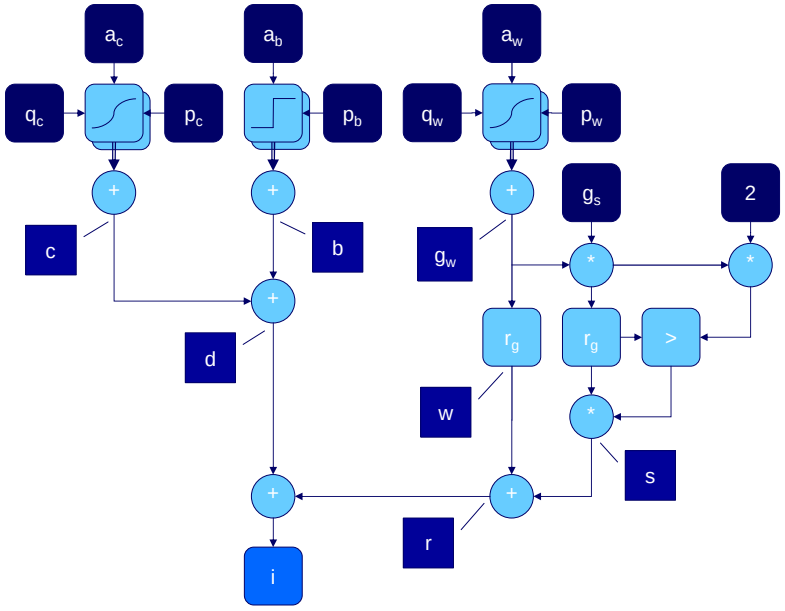

Figure 6.4: Indicator model overview.

Figures 6.5 to 6.10 show synthetically generated progressions representing normality, a maintenance action and a fault propagation. Each figure

is accompanied with the distributions from which their model parameters where chosen. All configuration parameters associated with steps and transitions are generated from random uniform distributions $r_u(from, to)$. For the counters K_w and K_c, the integer uniform distribution $r_{un}(from, to)$ is used, to that $\{K_w, K_c\} \in \mathcal{N}$.

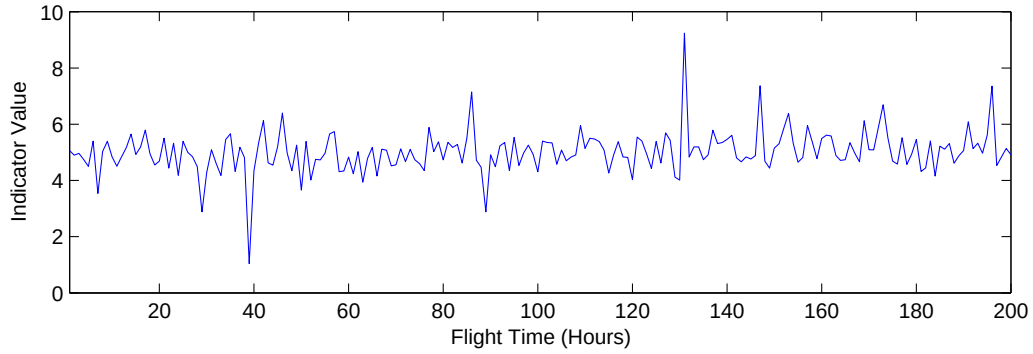

Figure 6.5: Normal indicator behavior.

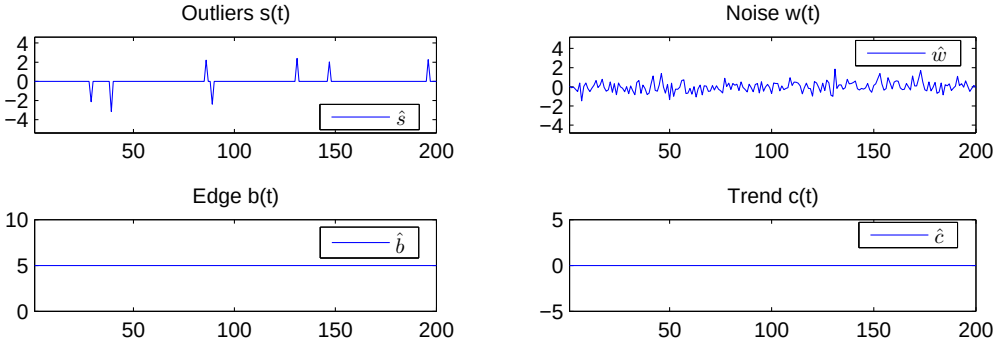

Figure 6.6: Normal progression components.

Symbol	Value	Symbol	Value
dc_w	0.5	dc_c	5
K_w	$\emptyset$	K_c	$\emptyset$
$a_w^{(k)}$	$\emptyset$	$a_c^{(k)}$	$\emptyset$
$q_w^{(k)}$	$\emptyset$	$q_c^{(k)}$	$\emptyset$
$p_w^{(k)}$	$\emptyset$	$p_c^{(k)}$	$\emptyset$
$g_s(t)$	$2g_w(t)$	a_b	$\emptyset$
		p_b	$\emptyset$

Table 6.1: Normal state model parameters

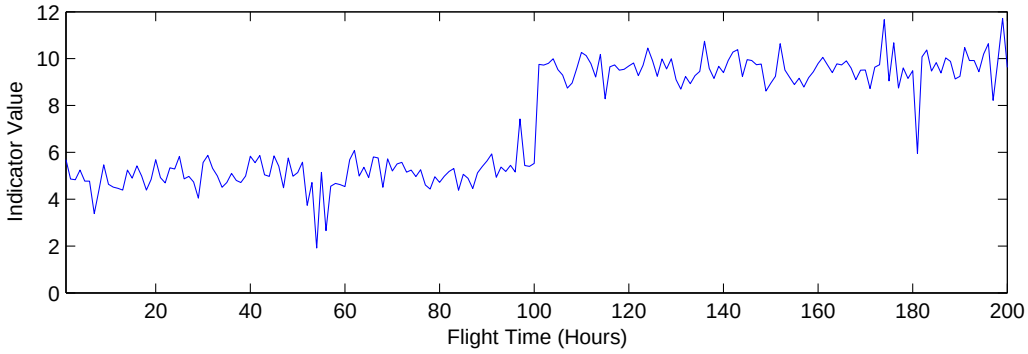

Figure 6.7: Step change due to maintenance.

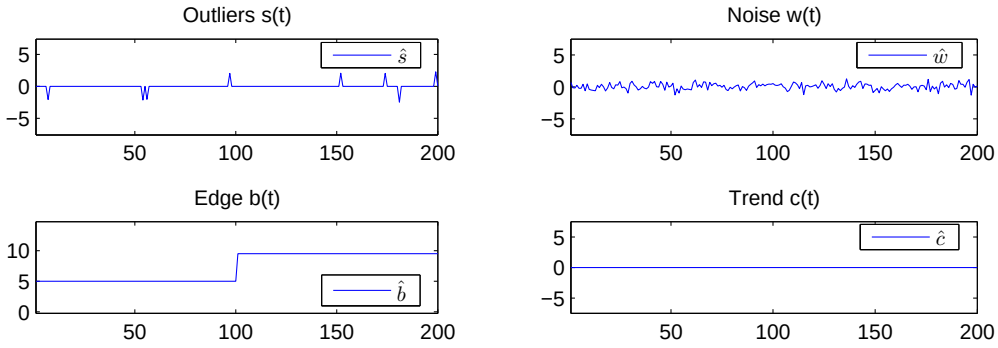

Figure 6.8: Step progression components.

Symbol	Value	Symbol	Value
dc_w	0.5	dc_c	5
K_w	$\emptyset$	K_c	$\emptyset$
$a_w^{(k)}$	$\emptyset$	$a_c^{(k)}$	$\emptyset$
$q_w^{(k)}$	$\emptyset$	$q_c^{(k)}$	$\emptyset$
$p_w^{(k)}$	$\emptyset$	$p_c^{(k)}$	$\emptyset$
$g_s(t)$	$2g_w(t)$	a_b	$r_u(1,4)$
		p_b	100

Table 6.2: Step change model parameters.

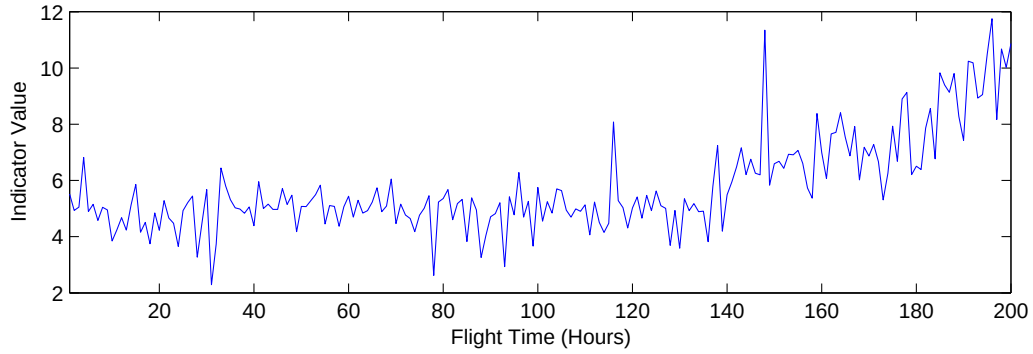

Figure 6.9: Trend due to component degradation.

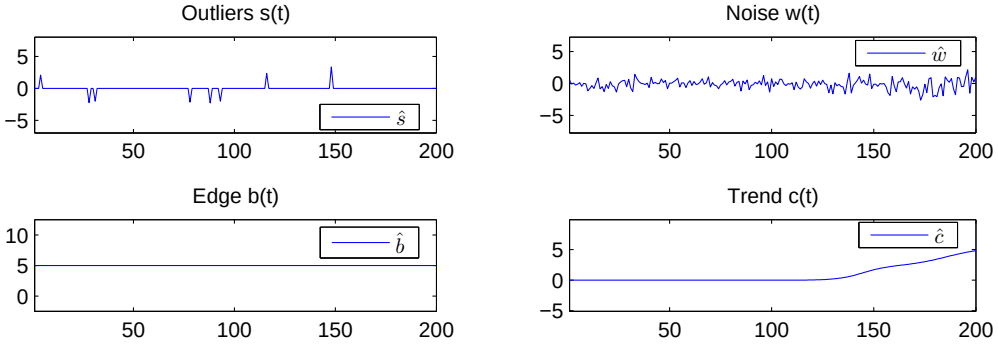

Figure 6.10: Trend progression components.

Symbol	Value	Symbol	Value
dc_w	0.5	dc_c	5
K_w	$r_{un}(1,4)$	K_c	$r_{un}(1,4)$
$a_w^{(k)}$	$r_u(0,1)$	$a_c^{(k)}$	$r_u(-5,5)$
$q_w^{(k)}$	$r_u(0.1,0.2)$	$q_c^{(k)}$	$r_u(0.2,0.5)$
$p_w^{(k)}$	$r_u(100,200)$	$p_c^{(k)}$	$r_u(100,200)$
$g_s(t)$	$2g_w(t)$	a_b	$\emptyset$
		p_b	$\emptyset$

Table 6.3: Damaged state model parameters.

The following three sections describe three methods to "reverse engineer" the component sum generated by the indicator model into its four base component. All three methods where tested on the same ten indicator series. The series have the same progression parameters for the model (Tab. 6.4), but with different seeds for the random number generator. Each method is illustrated with one of the progression from the test set. The component sum for this progression is given in (Fig. 6.11).

Symbol	Value	Symbol	Value
dc_w	0.5	dc_c	5
K_w	$r_{un}(1,4)$	K_c	$r_{un}(1,4)$
$a_w^{(k)}$	$r_u(0,1)$	$a_c^{(k)}$	$r_u(-5,5)$
$q_w^{(k)}$	$r_u(0.1,0.2)$	$q_c^{(k)}$	$r_u(0.2,0.5)$
$p_w^{(k)}$	$r_u(100,200)$	$p_c^{(k)}$	$r_u(100,200)$
$g_s(t)$	$2g_w(t)$	a_b	$r_u(1,4)$
		p_b	50

Table 6.4: Test set model parameters.

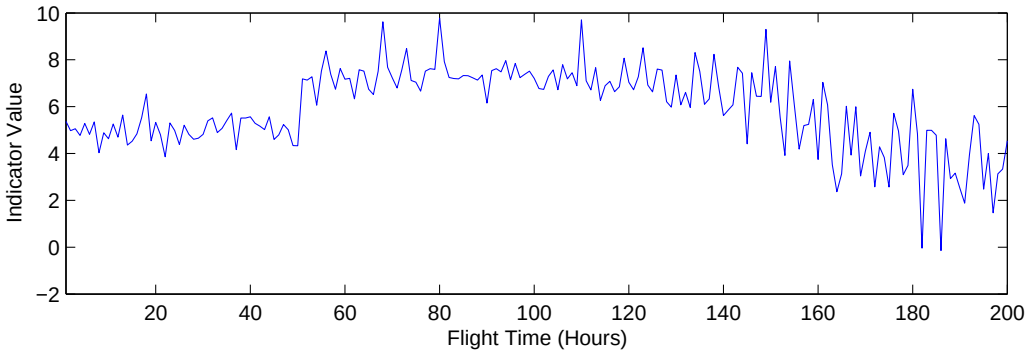

Figure 6.11: Progression generated by the configuration (Tab. 6.4)

6.3 Linear Progression Analysis

Before proceeding with fault detection, it is necessary to identify variations in the b, c, w and s components of which an indicator time series consist. A very interesting method developed by Sylvie Charbonnier et al. [5] permits dividing a time series into linear segments without the use of non-linear optimization. This section uses a variant of this algorithm as an alternative to the sigmoid based method developed in the previous section.

6.3.1 Segmentation

The segmentation algorithm starts by finding a first order polynomial approximation $\hat{d}_0$ of the first L_{init} samples of the dataset i. A cumulative error metric (Eq. 6.10) is used for validating $\hat{d}_0$ as an approximation of i. The dataset is then tested sample by sample starting at the beginning of the dataset. When e breaches the threshold T_{h1}, a marker is set in the dataset. Once e breaches a second threshold T_{h2}, $\hat{d}_0$ is rejected. A new first order polynomial approximation, d_1, is then estimated using the data from the marker up to the point where d_0 was rejected. This process is repeated until the entire dataset is segmented.

$$e(n) = \left| \sum_{k=0}^{n} d_0(k) - i(k) \right| \tag{6.10}$$

6.3.2 Segment Concatenation

The above segmentation process approximates a dataset i as a set $\hat{d}_k$ of linear segments which need not be continuous. In order to reduce the number of discontinuities in the approximation, the value at the beginning of each segment $\hat{d}_k$ is compared to the value at the end of it preceding segment $\hat{d}_{k-1}$. If this difference supersedes the threshold T_{hc}, the segments are left discontinuous. Inversely, if the gap between the segments is less than T_{hc}, the angle of $\hat{d}_k$ is changed so that its starting point matches the ending point of $\hat{d}_{k-1}$. This process removes minor discontinuous in the approximation. The component $\hat{d}$ is then constructed by concatenating all the $\hat{d}_k$ segments.

A noise estimate $\hat{r}$ is produced by subtracting $\hat{d}$ from the original observations (Eq. 6.11).

$$\hat{r}(t) = i(t) - \hat{d}(t) \tag{6.11}$$

Once $\hat{r}$ is obtained, its gain $\hat{g}_r$ is estimated using a sliding window rms (Eq. 6.12).

$$\hat{g}_r(t) = wrms(\hat{r}, t, L_r) \tag{6.12}$$

The components $\hat{w}$ and $\hat{s}$ are then separated by comparing each value in $\hat{r}$ to its gain estimate $\hat{g}_r(t)$. Any points being larger than T_s standard deviations of $\hat{r}$ are considered to be part of $\hat{s}$ (Eq. 6.13 and 6.14).

$$\hat{s}(t) = \hat{r}(t).(\frac{|\hat{r}(t)|}{\hat{g}_r(t)} > T_s) \tag{6.13}$$

$$\hat{w}(t) = \hat{r}(t) - \hat{s}(t) \tag{6.14}$$

Once $\hat{w}$ is obtained, its gain $\hat{g}_w$ is estimated using a sliding window rms (Eq. 6.15). A parametric model $\hat{\hat{g}}_w$ of the noise gain is obtained using the same parameterization methods that was used to identify $\hat{d}$.

$$\hat{g}_w(t) = wrms(\hat{w}, t, L_w) \tag{6.15}$$

6.3.3 Trend Analysis

An estimate for d is generated by the parameterization process. The component $\hat{b}$ is identified from the segmentation algorithm. Any two segments left discontinuous constitutes a change in $\hat{b}$, with position and amplitude given by the position and amplitude of the discontinuity. The component $\hat{c}$ is identified by subtracting $\hat{b}$ from $\hat{d}$. In order to detect changes in the condition of the underlying asset, fluctuations in the value of c and the gain of w must be monitored. This is done by computing the time derivative of $\hat{c}$ (Fig. 6.12) and $\hat{\hat{g}}_w$ (Fig. 6.13); $a_{\hat{c}}$ (Fig. 6.14) and $a_{\hat{\hat{g}}_w}$ (Fig. 6.15).

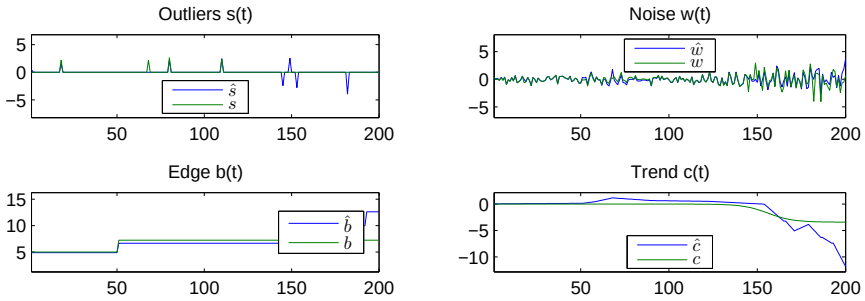

Figure 6.12: Indicator decomposition.

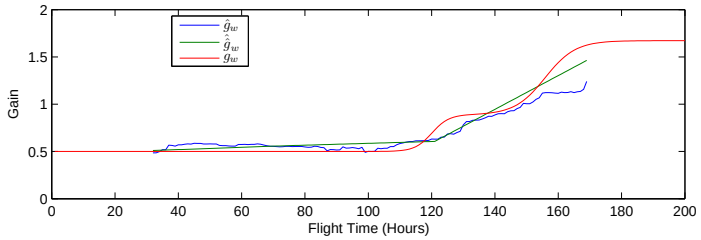

Figure 6.13: Indicator noise gain estimate.

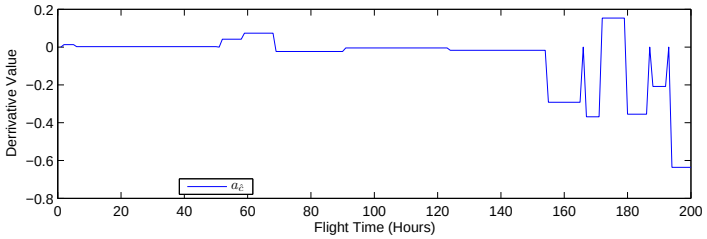

Figure 6.14: Indicator trend slope.

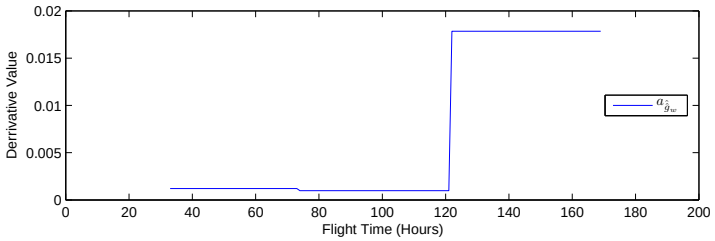

Figure 6.15: Indicator noise gain slope.

6.4 Sigmoid Progression Analysis

Although the line-based method performs very well, it is desirable to find a model which permits modeling of curved shapes. According to the progression model defined in the previous section, d is either constant, abruptly changing or gradually changing. When estimating the model $\hat{d}$ based on a set of observations of i, it is necessary to find a model prototype capable of assuming any behavior exhibited by d. A possible candidate is the sigmoid, as it is the primitive already used for generating gradual transitions in d. Further, it is also, with the correct set of configuration parameters, capable of producing abrupt transitions and straight lines.

The shape of the sigmoid prototype is adjusted by entry level dc_d, transition amplitude a_d, transition slope q_d, and transition point p_d, as illustrated by equation (Eq. 6.16) and figure 6.16 [51].

$$\hat{d}(t) = dc_d + \frac{a_d}{1 + e^{-q_d(t-p_d)}} \tag{6.16}$$

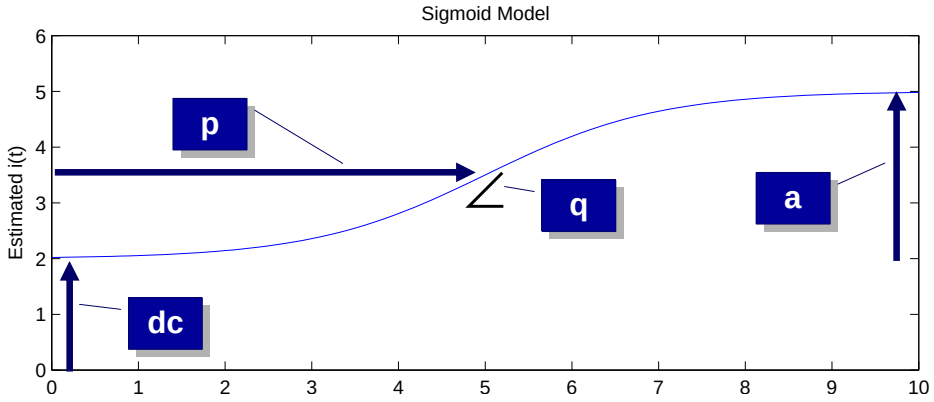

Figure 6.16: Sigmoid prototype.

Subtracting $\hat{d}$ from the observed indicator time series i obtains a scatter and outlier estimate $\hat{r}$ (Eq. 6.17). As r by definition is Gaussian noise, an $\hat{r}$ with a balanced power spectrum indicates that the model $\hat{r}$ and consequently the model $\hat{d}$ is correct. By adjusting the sigmoid shape parameters so that $\hat{r}$ assumes a white power spectrum, $\hat{d}$ assumes a close approximation of d.

$$\hat{r}(t) = i(t) - \hat{d}(t) \tag{6.17}$$

Once $\hat{r}$ is obtained, its gain $\hat{g}_r$ is estimated (Eq. 6.18) using a sliding window rms (Sec. A.2) of length L_r.

$$\hat{g}_r(t) = wrms(\hat{r}, t, L_r) \tag{6.18}$$

The components $\hat{w}$ and $\hat{s}$ are then separated by comparing each value in $\hat{r}$ to its gain estimate $\hat{g}_r(t)$. Any points being larger than T_s standard deviations of $\hat{r}$ are considered to be part of $\hat{s}$:

$$\hat{s}(t) = \hat{r}(t).(\frac{|\hat{r}(t)|}{\hat{g}_r(t)} > T_s) \tag{6.19}$$

$$\hat{w}(t) = \hat{r}(t) - \hat{s}(t) \tag{6.20}$$

Once $\hat{w}$ is obtained, its gain $\hat{g}_w$ is estimated using a sliding window rms (Eq. 6.21). A parametric model $\hat{\hat{g}}_w$ of the noise gain estimate is then obtained by adjusting the sigmoid parameters (Eq. 6.22) so that the sum square difference between $\hat{g}_w$ and $\hat{\hat{g}}_w$ is minimized.

$$\hat{g}_w(t) = wrms(\hat{w}, t, L_w) \tag{6.21}$$

$$\hat{\hat{g}}_w(t) = dc_w + \frac{a_w}{1 + e^{-q_w(t-p_w)}} \tag{6.22}$$

6.4.1 Sigmoid Series

This method can only model an indicator time series consisting of a single transition, i.e. a single trend or a single step change, and a single change in noise gain. A useful feature extraction algorithm must be sufficiently robust to be able to analyze an indicator time series consisting of several transitions. A solution is to model each transition in the indicator series with a separate sigmoid. This can be achieved by using a model with an arbitrary number of sigmoids (Eq. 6.23) and (Eq. 6.24).

$$\hat{d}(t) = dc_d + \sum_{k \in K_d} \frac{a_d^{(k)}}{1 + e^{-q_d^{(k)}(t-p_d^{(k)})}} \tag{6.23}$$

$$\hat{\hat{g}}_w = dc_w + \sum_{k \in K_w} \frac{a_w^{(k)}}{1 + e^{-q_w^{(k)}(t-p_w^{(k)})}} \tag{6.24}$$

The choice of model order K_d and K_w is discussed in the following.

6.4.2 Estimation Methods

The sigmoid sum model proposed above is underdetermined and non-linear, and cannot be estimated by matrix inversion like polynomial models. The problem of finding the set of model parameters which causes the model $\hat{d}$ to obtain a the best possible approximate of d is a non-linear optimization problem. A solution to this problem is finding the set of model parameters which generates the most balanced power spectrum for $\hat{r}$. Provided that the model order is fixed, this can be simplified to the problem of finding the set of model parameters which minimizes the $\hat{r}$ sum of squares [48]. However, if model order is itself a model parameter, minimizing square sum $\hat{r}$ will estimate a model $\hat{d}$ generating an $\hat{r}$ equal to zero. I.e. $\hat{d}$ models both d and r.

For the noise gain parametrization, it is sufficient to minimize the sum square difference between $\hat{g}_w$ and $\hat{\hat{g}}_w$. If model order is itself a parameter, it is however not possible to validate the model by evaluating the residual power spectrum. This because the sum square difference between $\hat{g}_w$ and $\hat{\hat{g}}_w$ is expected to assume a power spectrum of mainly low frequency, even for a correct model. Consequently, a traditional model validation technique like r^2 or adjusted r^2 must be used.

Several methods exist for non-linear optimization problems, two of them being evolutionary optimization and Trust Region. Evolutionary optimization is inspired by the process of natural selection. Trust Region is a traditional steepest descent method.

All methods presented here attempt to minimize square sum $\hat{r}$, unless otherwise stated. The two former methods need model order to be decided in advance, while the remaining ones are capable of estimating this parameter. All methods were tested using a synthetically generated dataset.

Evolutionary Optimization with Pre-Defined Order (EO)

This methods uses evolutionary optimization (Sec. A.4.2) to adjust the sigmoid parameters given a predefined function order, with the aim to minimize sum square $\hat{r}$. As the function order is fixed, there is no danger of "over fitting". The method was tested using an elite ratio of 0.1. Of the remaining individuals, the crossover fraction was set to 0.8 and mutation fraction the remaining 0.2. Initial range for dc_d and the $a_d^{(k)}$ parameters were set to the range of i. All $p_d^{(k)}$ parameters had their initial range set to the range of t, and the $q_d^{(k)}$ parameters where given the static initial range from 0 to 10. A population size of 200 individuals where evaluated over 200 generations with the end result shown in figure 6.17. Orders was set to 3.

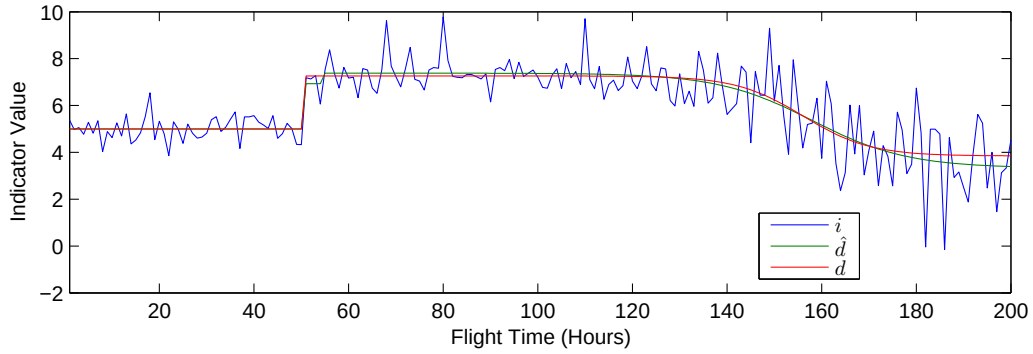

Figure 6.17: Calibration set with approximated using evolutionary optimization.

For the example dataset, the approximation has one miss-placed step and a significant bias toward the end of the dataset. In general, this method works well. The main drawbacks is that it does not determine order, and convergence is slow due to a poorly chosen initial condition.

Trust Region with Pre-Defined Order (TR)

This method was tested using convergence of the solution to the approximate function as stopping criteria. The dc_d parameter was given the initial value equal to μ_i, while the $a_d^{(k)}$ and $q_d^{(k)}$ parameters were set to 0 and 1 respectively. Each sigmoid's position, $p_d^{(k)}$, was set so that the sigmoids were uniformly distributed across t. Final results are shown in figure 6.18. Orders was set to 3.

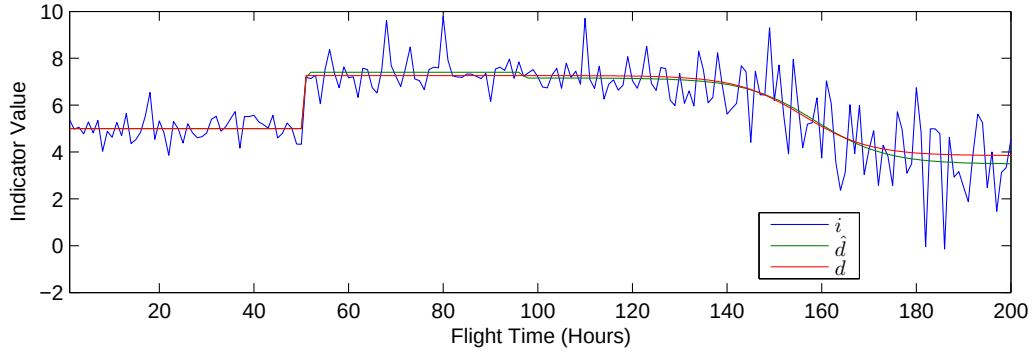

Figure 6.18: Calibration set with approximated using Trust Region with simple initial guess.

This method performs reasonably well on the example dataset. The main

objection to this method is execution speed, as robust evolutionary optimization requires a large number of evaluations of the object function. Further, this method is incapable of determining model order.

Residual Spectrum Validation (RSV)

In the context of HUMS data analysis, it would be sufficient to analyze the last 200 - 300 flight hours of an aircraft to assess the condition of the drive-train. For an indicator series of this duration, more than three or four transition would be extremely unlikely. Consequently, good results can be obtained using a fixed model order. Still it would be desirable to automatically determine the optimal model order, with the optimal order being one which balances the power spectrum of $\hat{r}$ while at the same time minimizes model order.

A simple way of determining the best model order is to start with a first order model. For most realistic cases, this will be sufficient. The noise estimate is then evaluated by the function J (Eq. 6.25).

$$J = \frac{(\sum_{n=0}^{N-1} \hat{r}(n)^2)^2}{\sum_{n=0}^{N-1} \sum_{k=-\infty}^{\infty} \hat{r}(n)\hat{r}(n-k)} \tag{6.25}$$

As d is a low frequency process, any contribution from d in $\hat{r}$ will be confined to the first few DFT coefficients, and will thus unbalance the otherwise white spectral content of $\hat{r}$. The function J determines the whiteness of $\hat{r}$. This function has an expected value of 0.5 for white noise, and less than 0.5 for non-white signals. If $\hat{r}$ is not sufficiently white, the model order is incremented and the model parameters reestimated. This process is repeated until an acceptable model is found.

The whiteness acceptance criterion is more suitable for this application than traditional goodness-of-fit criteria, like r^2 or adjusted r^2, as these are energy-based metrics. In this context, one can however not make any assumption about the energy distribution between d and r. The only identifying mark remains the power spectrum of r.

This approach can be implemented as a wrapping around the two previous methods, thus identifying both the optimal model order and the optimal value for each parameter. The method was tested using Trust Region and a spectral acceptance criterion of 0.45, with final results in figure 6.19. Spectral validation produces repeatedly good results when used with Trust Region. Execution time is low, but the method seems inherently robust.

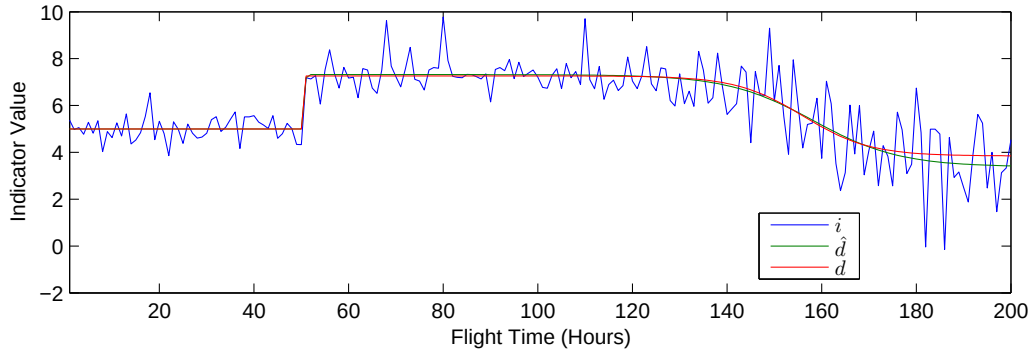

Figure 6.19: Calibration set with Trust Region residual spectrum validation model.

Iterative Evolutionary Estimation (IEO)

Although evolutionary optimization is a robust method in the presence of local optimums, the algorithm struggles when the parameter space grows to waste. In order to maintain robustness, the algorithm should maintain a number of individuals sufficient to span the space containing the optimal solution with a certain population along each dimension. If for instance a one dimensional problem is solved using a population of k individuals, then a two dimensional problem will require k^2 individuals to maintain the same population density. This becomes a problem when the order of the sigmoid model increases, as this causes the number of dimension in parameter space to increase three times as fast, thus causing the number of individuals necessary to maintain population density to increase exponentially.

The problem of approximating an indicator series containing several transitions is one where a problem consisting of several sub problems is more complex than the sum complexity of the sub problems. This because an N^{th} order model can have $N!$ different orderings of its sigmoids which for the optimization algorithms are seen as $N!$ different solutions, even though they indeed are equivalent.

A simplification will thus be to solve one sub-problem at a time, instead of trying to solve all the problems at once. This can be attempted by splitting the input series into uniform segments, and approximating one segment at a time. The algorithm starts by estimating a first order model to the first K observations of the input series. It then extends the estimation window by K observations, and estimates a model consisting of two sigmoids, treating the previously estimated *dc* component as static. I.e. the estimate for the *dc* component is kept while the sigmoid is re-estimated. The estimation window is extended by another K points for the third iteration, and the *dc* component

and first of the sigmoids are frozen while the model is re-estimated using an additional two sigmoids. This process, estimating each sigmoid two times, continues until the length of the estimation window reaches the length of i.

This method was tested with the same configuration as the previous evolutionary method, but using 50 individual for 50 generations at each iteration. Final results are shown in figure 6.20.

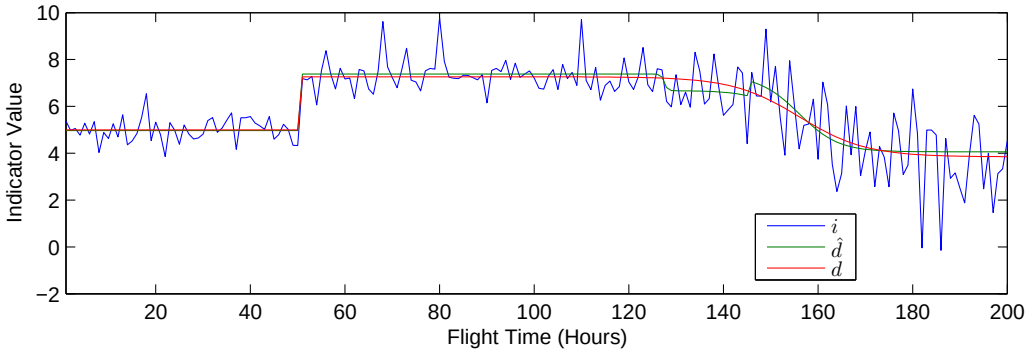

Figure 6.20: Calibration set with iteratively approximated sigmoid model.

This method should be seen as a failure. It is exceptionally slow, and produces erratic results. A main reason for its lack of precision is that it does not solve the global optimization problem. This is visible in the output, like in the example dataset, where output behavior is abruptly changing between segments.

Trust Region using Band-Limited Differentiator Pre-Processing

The success or failure of a gradient search optimization is to a large extent given by the initial guess, i.e. the initial current position in parameter space. Consequently, if a more accurate initial guess can be made, robustness and convergence speed will increase.

If the position and amplitude of each significant transition can be approximated in a linear manner, then only transition slope need to be estimated through non-linear optimization. A common approach to trend detection is using band-limited differentiators [9].

Due to the high levels of noise, the derivation filter is applied to a de-noised version of i, using noise removal methods developed in [12] and further adapted to health indicators. This de-noising method has the advantage, compared to simply altering the bandwith of the derivation filter, of preserving edges while retaining a good damping of white noise. By detecting zero-crossing of the first derivative, local optimums along the time-series are

discovered. Transition points can then be set in between each optimum. The corresponding transition amplitudes are given by the amplitude difference in the de-noised i between the leading and trailing optimum to each transition point. Initial approximate for dc is the mean value from the start of the series to the first zero-crossing. Final results are shown in figure 6.21.

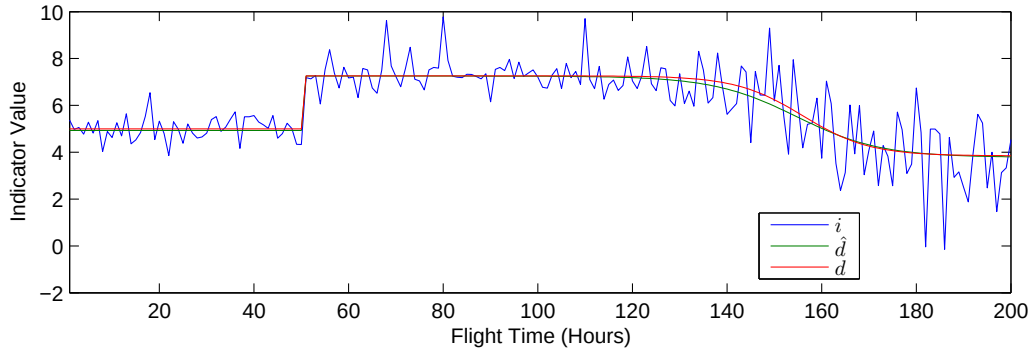

Figure 6.21: Calibration set with sigmoid approximate.

This method produces repeated good results, and has the lowest execution time of all the models. Approximation is very close on the example dataset, as well as all other datasets on which it has been tested.

Comparison

Although execution speed and model error for the different methods can easily be compared, which algorithm best model the features of operational importance remains a subjective opinion. Execution time is of course of interest, but as the analysis of HUMS data is done off-line, this point is not crucial. The main point of interest is in each algorithm's robustness in modeling the operationally important features correctly for any realistic input series. For such a study, the test-sets used here are not sufficient. Consequently, identifying the optimal method becomes a somewhat subjective choice.

The two initial methods require order to be determined in advance. Although, in an industrial setting, such a requirement can be met by using a reasonable order given the length of the data set, it is also desirable to have a more autonomous solution. Further, the evolutionary algorithm is not well suited for high order models.

The spectral validation method produces good results when used as a wrapping around Trust Region. An objection to this algorithm is execution time, as it re-estimates the model each time order is incremented, thus requir-

ing substantial execution time for high order models. This could probably be amended by using the final results from the last optimization as initial condition when model order is incremented, simply adding a sigmoid in the region where model error is largest. Such improvements have however not been further explored in this study.

Iteratively evolutionary estimation was an attempt to estimate a complex function using a reduced parameter space, by estimating one segment at a time. The method is still slow, and lacks accuracy because it does not solve the global optimization problem.

Trust Region with initial guess given by band-limited derivation is clearly the best method for the problem at hand, as the initial position is very close to the optimal solution. This means that convergence is quick, and that the initial position is closer to the global optimum than any local optimums.

6.4.3 Trend Analysis

Given a $\hat{d}$ which is optimized to closely approximate d, the parameters controlling $\hat{d}$ constitutes a compact form representation of $\hat{d}$. By analyzing this compact form representation, it is possible to understand the behavior of $\hat{d}$, and the d which it approximates. The traversal through states associated with mechanical degradation is however manifested as changes in the value of c not d. It is thus necessary to separate the contribution from $\hat{b}$ and $\hat{c}$ in $\hat{d}$. As already explained, a step change constitutes a change in b while a trend constitutes a change in c. In the progression pattern referred to as normal, both b and c are static.

From this information, the key properties $a_r^{(k)}$, $a_d^{(k)}$ and $q_d^{(k)}$ are extracted. The two latter parameters hold the information necessary to identify what type of transition is being approximated by a sigmoid, and consequently the behavior of b and c within the region covered by the sigmoid. The former parameter holds the information necessary to understand the progression of the indicator scatter level.

In order to separate c and b, the $q_d^{(k)}$ metric is tested against a threshold T_q. If a $q_d^{(k)}$ value overshoots this threshold, transition k is considered to be part of b. Inversely, if a $q_d^{(k)}$ value is inferior to T_q, transition k is considered to be part of c. After the transition has been sorted, $\hat{b}$ and $\hat{c}$ are constructed from their respective sigmoids, giving the full 4-way decomposition(Fig. 6.22). In order to detect changes in the condition of the underlying asset, fluctuations in the value of c and the gain of w must be monitored. This is done by computing the time derivative of $\hat{c}$ (Fig. 6.22) and $\hat{\hat{g}}_w(t)$ (Fig. 6.23); $a_{\hat{c}}$ (Fig. 6.24) and $a_{\hat{\hat{g}}_w}$ (Fig. 6.25).

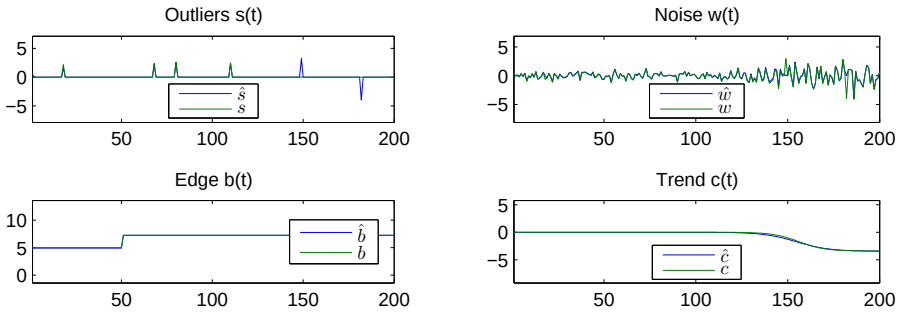

Figure 6.22: Indicator decomposition.

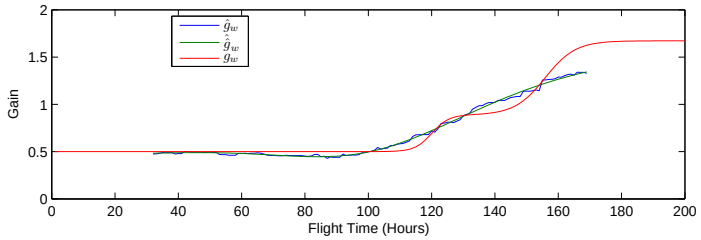

Figure 6.23: Indicator noise gain estimate.

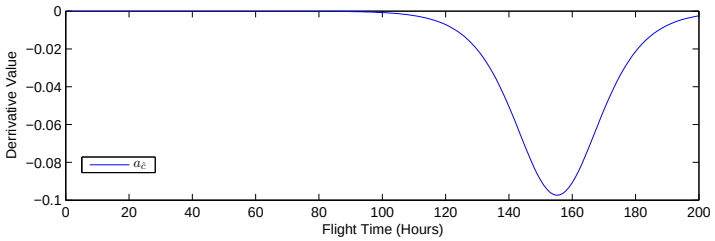

Figure 6.24: Indicator trend slope.

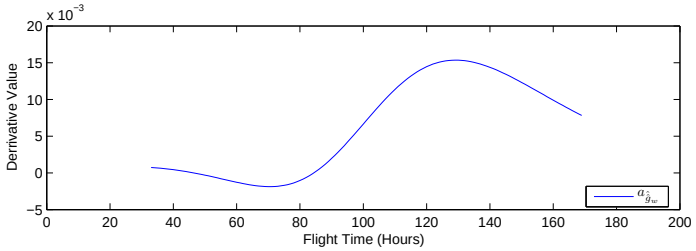

Figure 6.25: Indicator noise gain slope.

6.5 Non-Parametric Progression Analysis

The trend analysis model developed in the previous section was based on the model believed to generate the observable time series. Although choosing such an analysis model permits estimating an accurate approximate of the original data, model estimation is difficult due to the non-linear nature of the model. It is thus desirable to find a trend analysis method not in need of non-linear optimization techniques. This is achievable through the use of band-limited differentiators [9]. The method developed in this section is based on the same principle as band-limited differentiators, but is further specialized to fit the characteristics of condition indicator time series [52].

In the previous section, it was assumed that the observed indicator time series are the sum of a deterministic process and a random noise process. Like in the previous section, $i(n)$ is split in four components; an outlier process $s(n)$, a random noise process $w(n)$, an edge process $b(n)$, and a trend process $c(n)$.

These components correspond exactly to the component used in the synthetic indicator progression model. The traversal through states associated with mechanical degradation is as already explained manifested as changes in the value of c and the gain of w. A first step in the fault detection process is thus to separate these four components.

6.5.1 Outlier Separation

The dataset i is de-trended (Eq. 6.26) by having its moving median (Sec. A.1) at window size L_{s-mm} removed. This filter is an effective form of denoising, and will remove all of s and some of w, while keeping most of c and b intact. The modified dataset i'_1 will consequently contain all of s, some of w and very little of c and b.

$$i'_1(n) = i(n) - mm(i, n, L_{s-mm}) \tag{6.26}$$

As the modified dataset has very little trend or edge contribution, it will have zero mean. An outlier is defined as a point of value T_s standard deviation outside the mean of the dataset (Eq. 6.27). Windowed rms (Eq. A.2) at window size L_{s-wrms} is used as signal scatter might vary along the time line.

$$\hat{s}(n) = i'_1(n).(\frac{|i'_1(n)|}{wrms[i, n, L_{s-wrms}]} > T_s) \tag{6.27}$$

Before proceeding with the separation of w, b and c, the outlier component is removed from the dataset (Eq. 6.28).

$$i_1(n) = i(n) - \hat{s}(n) \tag{6.28}$$

6.5.2 Edge Separation

Wavelet expansions allow a signal x to be represented as a weighted sum of scalings and dilatations of a wavelet function $\psi(t)$. The Continuous Wavelet Transform (CWT) (Sec. A.3.1) calculates, given an input signal and a wavelet function, the weights corresponding to each scaling and dilatation of $\psi(t)$ (Eq. 6.29). This produces a matrix $cwc_x^{(j,n)}$ with the j index representing the scaling dimension and the n index representing dilatation or time. The $cwc_x^{(j,n)}$ matrix can be interpreted as a spectrogram, although the relationship between scale and frequency, in the Fourier sense, depends on the choice of wavelet.

$$cwc_{i_1} = cwt(i_1, \psi) \tag{6.29}$$

Edges are easy to spot in the dataset, and are usually synonyms with maintenance actions. In order to detect edges, the indicator series is expanded on the Haar [19] wavelet using at scales 1 through J_{edge}. The meaning of a wavelet coefficient matrix depends on the choice of the wavelet. For the Haar wavelet, the coefficients signify the numeric derivative of the dataset at different scales. I.e. the vector $cwc_{i_1}^{j}$ contains the dataset mean derivative across a sliding window of 2^j points. The Haar wavelet is chosen because it resembles a step. Thus, whenever a step in encountered in the dataset, the wavelet coefficients will exhibit higher values than if no step is present.

Trends are slowly evolving phenomena, and are thus confined to the coarser scales of the cwc_{i_1} matrix. Random noise is wide band, but its presence in the coarse scales is negligible compared to the energy of the trends. The only component with a significant impact across all scales is the edge. The effect of a unit step at a given scale is $2^{\frac{j}{2}}$.

Consequently, an edge can be identified by looking for the edge signature across the scales. A modified detail matrix $cwc_{i_1}^{\prime(j,n)}$ (Eq. 6.30) is created to capture the amplitude of the edge. An edge at position n will produce a modified detail matrix with coefficients $cwc_{i_1}^{\prime(j,n)}$ equal to the edge amplitude for all values of j along the n'th column.

$$cwc_{i_1}^{\prime(j,n)} = 2^{\frac{-j}{2}} cwc_{i_1}^{(j,n)} \tag{6.30}$$

Using the above definition, an edge is a position in time n where $cwc_{i_1}^{\prime(j,n)}$ is equal for all values of j. Due to the presence of components w and c,

the values across the scales will not be completely identical. Thus, an edge signature metric is defined (Eq. 6.31).

$$b_p(n) = \frac{|mean[cwc_{i_1}^{\prime(j,n)}]|}{std[cwc_{i_1}^{\prime(j,n)}]} \tag{6.31}$$

This represents the degree of edge behavior in each point n along the time line. The functions *mean* and *std* are computed across all scales j for each point n in time. Thus, an edge can be defined as a point in time n where $b_p(n)$ is higher than the threshold T_p. Unfortunately, this will also capture minor transition which also satisfies the above criteria. Thus, an edge magnitude criteria is therefore introduced (Eq. 6.32).

$$b_m(n) = \frac{|mean[cwc_{i_1}^{\prime(j,n)}]|}{wrms[i_1, n, L_b]} \tag{6.32}$$

An edge is a transition which rises clearly above the background noise. The above equation will normalize the edge magnitude by the total dataset energy in a trailing window with size L_b. An edge exists in a point in time n which satisfies the trend signature criteria, while also having a magnitude $b_m(n)$ larger than T_m (Eq. 6.33).

$$b_{true}(n) = (b_p(n) > T_p) \wedge (b_m(n) > T_m) \tag{6.33}$$

A recursive equation (Eq. 6.34) provides an estimate for the edge process. This equation always outputs it last value except when a step is detected, in which case it adds the amplitude of the step to its output value.

$$\hat{b}'(n) = b'(n-1) + mean[cwc_{i_1}^{\prime(j,n)}].b_{true}(n) \tag{6.34}$$

The initial value of $\hat{b}'$ is zero. A modified version, $\hat{b}$, will be developed later. The initial value of this component will be the initial value of the dataset, after s and w are removed. Another modified dataset, i_2, is constructed without the edge component.

$$i_2(n) = i_1(n) - \hat{b}'(n) \tag{6.35}$$

6.5.3 Random Noise Separation

In order to perform a CWT which contains all information about the source signal, the source signal must be analyzed at an infinite number of scales, making reconstruction impossible. This problem is overcome by the Discrete Wavelet Transform (DWT)(Sec. A.3.2) and the Stationary Wavelet

Transform (SWT) (Sec. A.3.3), which expands the input signal on a wavelet function using an arbitrary number of scales. The remainder of the signal, which can not be expanded using the number of scales chosen, is left in the approximate vector. Consequently, the approximate vector *swta* and the detail matrix *swtd* will together contain all information in the original signal, making reconstruction possible. While the DWT has applications in signal compression, the SWT (Eq. 6.36) is the preferred choice for de-noising.

$$[swta_{i_2}, swtd_{i_2}] = swt(i_2, \psi, J_{noise}) \tag{6.36}$$

This study deals only with finite signals. In order to keep transform output length the same as input length, while reducing transients, signals / coefficients are padded at start and end. Padding consists of samples having the mean value of the K first / last samples. This provides better results than periodic padding, circular convolution, as the start and end of the signals used in this study can have very different amplitudes.

The dataset i_2 is expanded on the db5 wavelet [19] using the SWT at scales 1 through J_{noise} (Eq. 6.36). The constant J_{noise} is chosen, for a realistic dataset, to capture most of the trend energy in $swta_{i_2}$. Regardless of the trend distribution between $swtd_{i_2}$ and $swta_{i_2}$, the vector $swtd_{i_2}^{(1,n)}$ has virtually no contribution from c. Consequently, the energy in $swtd_{i_2}^{(1,n)}$ is only w. Assuming w to be Gaussian white noise, the energy level in $swtd_{i_2}^{(1,n)}$ is representative for the contribution of w across all scales. Using the windowed RMS, a w energy estimate across time is made.

$$\hat{g}_w(n) = wrms[swtd_{i_2}^{(1,n)}, n, L_w] \tag{6.37}$$

The component w is then assumed to be the coefficients in $swtd_{i_2}$ with absolute value less than T_w. As w is assumed to be white, the same threshold is applied across all scales.

$$swtd_w^{j,n} = swtd_{i_2}^{j,n}.(|swtd_{i_2}^{j,n}| < \hat{g}_w(n).T_w) \tag{6.38}$$

Standard de-noising usually consists of setting the smallest *swtd* coefficients to zero before reconstructing. As the purpose of this exercise is to capture the noise w rather than the signal c, the largest $swtd_{i_2}$ coefficients and all of $swta_{i_2}$ are zeroed out before reconstruction.

$$\hat{w} = idwt(0, swtd_w^{j,n}, \psi) \tag{6.39}$$

The edge component is based on the b' calculated above, but corrected so that its initial value is the initial value of the dataset minus w and s.

$$\hat{b}(n) = \hat{b}'(n) + i_2(0) - w(0) \tag{6.40}$$

The trend component is the remaining data after s, w and b has been removed.

$$\hat{c}(n) = i(n) - \hat{s}(n) - \hat{w}(n) - \hat{b}(n) \tag{6.41}$$

Figure 6.26 shows the entire separation process as a flow chart, with input, output, and constants.

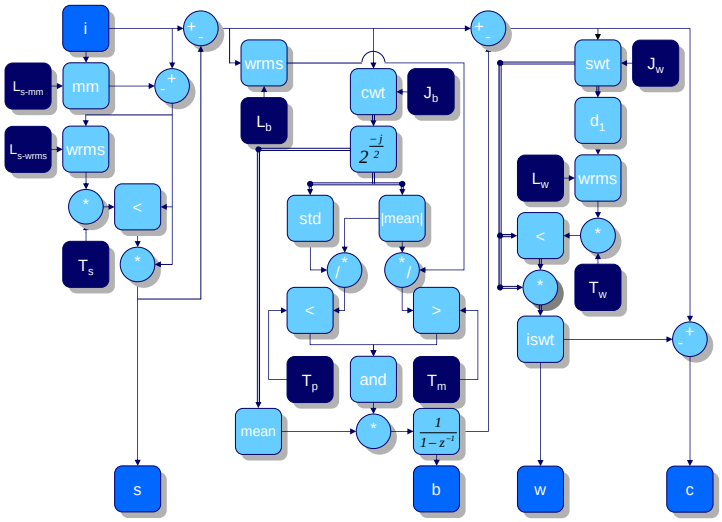

Figure 6.26: Source splitter overview.

6.5.4 Trend Analysis

As already stated, it is in the value of c and the gain of w that are of interest to uncover mechanical faults. Even though this algorithm manages to split the four components making up i, it does not produce a parametric model whose parameters can be evaluated to understand the behavior of the data. It is thus necessary to perform an additional parametrization step, in the form of a trend analysis of c as well as $\hat{g}_w$ from (Eq. 6.37).

The HUMS acquires data during flight from each sensor at regular intervals, so that the spacing between each indicator value, in flight time, is relatively uniform. All methods discussed here assumes uniform spacing. For datasets where this is not the case, with for instance missing data due to sensor problems etc., the indicator series must be interpolated with a smoothing-function and re-sampled.

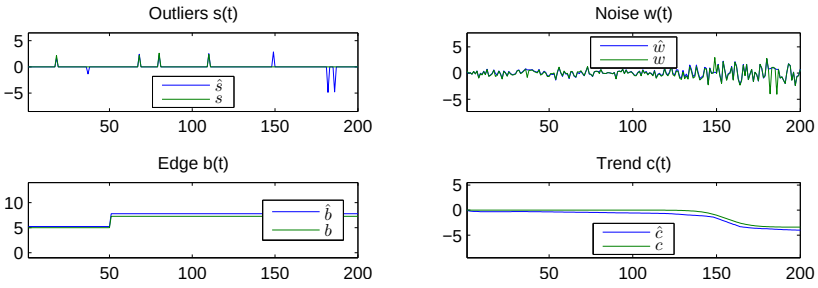

Figure 6.27: Indicator decomposition.

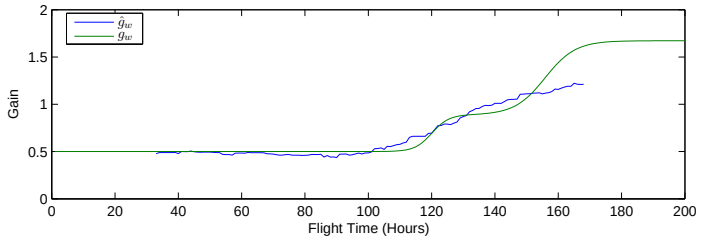

Figure 6.28: Indicator noise gain estimate.

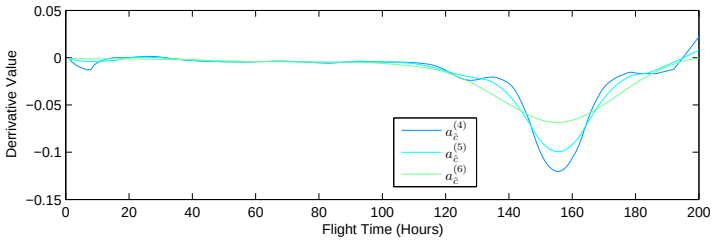

Figure 6.29: Indicator trend slope.

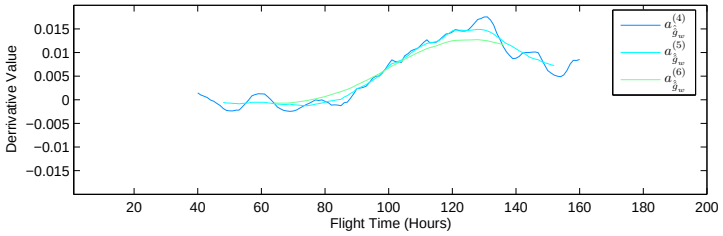

Figure 6.30: Indicator noise gain slope.

Trend analysis of a signal x is performed by the CWT using the the Haar wavelet [19]. This corresponds to a sliding window linear regression. Window size is given by the scale parameter j so that the size of the window in which the linear regression is performed equals 2^j. Consequently, a small value for j will capture rapid fluctuations, while large values for j captures longer trends. In order to detect the increasing and decreasing trends associated with mechanical degradation, it is necessary to use several values for j. This produces a coefficient matrix $a_c^{(j)}(n)$ with dimension N by J. Figures 6.29 and 6.30 shows how the different wavelet scales reacts to fast and slow movement in $\hat{c}$ (Fig. 6.27) and $\hat{g}_w$ (Fig. 6.28).

$$a_c^{(j)}(n) = cwt(c, \psi) \tag{6.42}$$

$$a_{\hat{g}_w}^{(j)}(n) = cwt(\hat{g}_w, \psi) \tag{6.43}$$

6.6 Calibration

Both the parametric and the non-parametric feature extractor depends on a number of tuning parameters to perform a correct decomposition of the input data. An optimal configuration of these parameters depends on the type of input, i.e. the energy distribution between s, w, b and c, and is essential for the performance of the algorithms.

The fundamental function of the feature extractor is to map an input vector to a set of output vectors. In the most general of terms, this is the same functionality provided by other non-linear mapping tools, like fuzzy logic classifiers and artificial neural networks. There are two main approaches for calibrating non-linear mapping tools; through expert knowledge, as normally applied to fuzzy logic systems, or through the use of training data.

Traditional training methods using marked training sets are inapplicable in this scenario, as the correct decomposition, i.e. the desired output, for a real time series cannot be known. Using expert knowledge, it is possible to derive the configuration parameters from a set of specifications describing behavior of the different signal components. Any such specification will however mostly be guesswork based on user experience. As an alternative approach, a synthetic dataset is generated using the model developed in the previous section. The correct decomposition of this dataset is known from the definition of the dataset, and can be used as a target for automatic training.

The four artificially generated components are added together before the source splitter algorithm is applied. The output from the source splitter, an estimate of the four components, is compared to the original components and

an error metric is produced. This procedure is used as object function for an evolutionary algorithm, set to find the optimal value for the configuration parameters.

An evolutionary algorithm runs 5 individuals per dimension for 20 generations with an elite ratio of 0.1 and a crossover fraction of 0.8. As the dataset contains stochastic elements, the optimization procedures maximize performance across 10 different datasets generated with different seed for the random number generator.

6.6.1 Linear Progression Analysis

The line based feature extractor requires three parameters; T_h1, T_h2 and T_hc [5]. In addition to this, it requires the same parameters as the sigmoid model to separate $\hat{s}$ and $\hat{w}$. This produces two optimization sets; $\{T_h1, T_h2, T_hc\}$ and $\{L_r, T_s\}$. The first problem is solved by finding the parameters that minimizes the square sum error for s. The second problem is solved by finding the parameters that minimizes the square sum error for d.

Figure 6.31 shows the synthetic components as well as the estimated ones using the configuration parameters obtained by the calibration procedure.

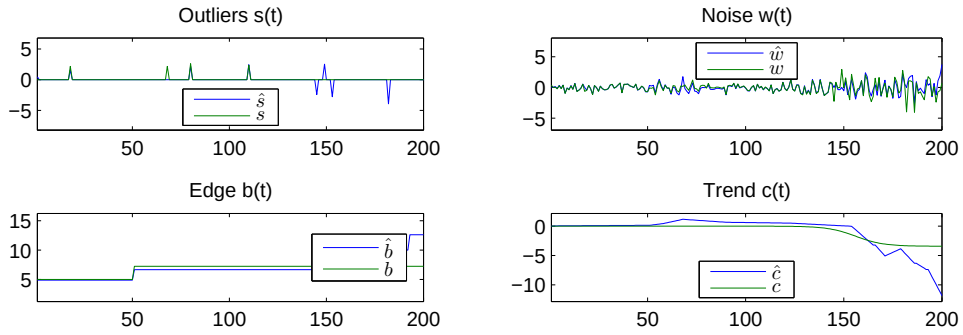

Figure 6.31: Decomposition of synthetically generated dataset.

6.6.2 Sigmoid Progression Analysis

The sigmoid feature extractor requires 3 configuration parameters to be set; L_r, T_s, T_q. The former parameter in $\mathcal{N}$, while the two latter are in $\mathcal{R}$.

To reduce the number of dimensions in the search space, the parameters controlling the splitting of $\hat{r}$ and $\hat{d}$ are estimated separately. This generates two optimization problems; $\{L_r, T_s\}$ and $\{T_q\}$. The first problem is solved by minimizing square sum error for s. The second optimization problem is solved by minimizing the number of miss-placed edges.

Figure 6.32 shows the synthetic components as well as the estimated ones using the configuration parameters obtained by the calibration procedure.

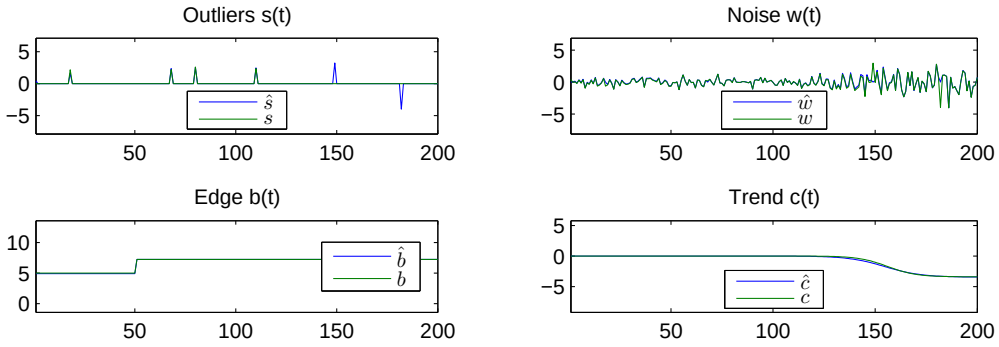

Figure 6.32: Decomposition of synthetically generated dataset.

6.6.3 Non-Parametric Progression Analysis

The non-parametric feature extractor requires 10 configuration parameters to be set; L_{s-mm}, L_{s-wrms}, L_b, L_w, J_b, J_w, T_s, T_p, T_m and T_w. The six former parameters are in $\mathcal{N}$, while the four latter are in $\mathcal{R}$.

To reduce the number of dimensions in the search space, the parameters controlling the estimation of s, b and w are estimated separately. This generates three optimization problems; $\{L_{s-mm}, L_{s-wrms}, T_s\}$, $\{L_b, J_b, T_p, T_m\}$ and $\{L_w, J_w, T_w\}$. The first and last problem, i.e. the estimation of the parameters controlling s and w, are solved by minimizing square sum error for s and w respectively. The second optimization problem, i.e. estimating the parameters for identifying edges, is solved by minimizing the number of miss-placed edges.

Figure 6.33 shows the synthetic components as well as the estimated ones using the configuration parameters obtained by the calibration procedure.

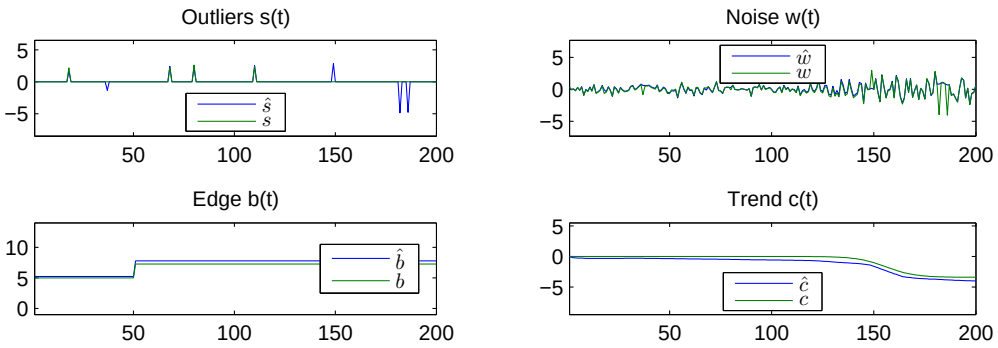

Figure 6.33: Decomposition of synthetically generated dataset.

6.7 Conclusion

Using synthetic datasets generated by the progression model, the sigmoid analysis methods outperforms easily the non-parametric analysis. This can however to a large extent be contributed to the fact that the parametric analysis method and the progression model uses the same progression primitive; the sigmoid. As there is no physical evidence that indicator progressions assume sigmoid shapes, other than that observable indicator trends look "sigmoid like", this might give the parametric method undeserved good results when looking only at approximation error. Two other points of interest is computing time and robustness. As the parametric method uses non-linear optimization to estimate its progression model, it is significantly slower than the line-based and non-parametric once and is not guaranteed to find the optimal solution. Although Trust Region with differentiator pre-processing significantly obtains both convergence speed and robustness, using non-linear optimization still poses a certain risk.

The linear method has the advantage of not requiring non-linear optimization. Looking exclusively estimation error, this method will in some cases produces large deviations between algorithm output $\hat{d}$ and the pre-defined target d. This can to some extent be contributed to the fact that this is a real-time algorithm. As the method must detect a significant change in indicator slope before it can adjust its output, it sometimes "overshoots" sudden changes by a few samples. Another reason for large estimation error on simulated data, compared to the sigmoid model, is that the sigmoid model and the simulated data uses the same primitive, giving undeserved good results. As far as the non-parametric method is is concerned, it will for the task of separating d and r largely emulate a lowpass filter, for whose performance is difficult to challenge.

Table 6.5 contains the $\hat{d}$ square sum error across ten test signals for each of the sigmoid methods, as well as the linear and the non-parametric method (NP). From this, it appears as if the non-parametric method is the most interesting. Evolutionary optimization and Trust Region using pre-defined order have slightly lower square sum error, but has the clear disadvantage of not optimizing function order. This makes the methods less reliable for signals with variable length and complexity. The non-parametric method is thus the favored progression analysis tool, as it produces repeated good results, and has superior computational speed.

Set	EO	TR	RSV	IEO	TRD	Line	NP
1	41.853	26.6643	33.4888	54.2973	37.3468	25.5291	23.6302
2	7.696	9.1261	9.2098	55.9464	9.3499	50.1149	10.2
3	7.9776	7.4386	54.6359	33.9158	56.9523	14.8669	13.3758
4	20.9424	20.1561	19.1953	73.9504	22.7163	124.2189	42.2564
5	7.641	14.9461	71.0944	94.8918	68.3208	72	11.7348
6	20.0249	23.8039	84.5186	64.5285	9.9265	67.8417	16.1921
7	27.3242	30.1642	57.9581	2550.4486	50.9997	60.3503	18.0125
8	19.3812	4.7417	4.7386	76.1091	36.4489	62.1104	17.3813
9	2.3191	9.2118	9.2118	18.9621	8.0787	34.2552	11.0039
10	20.6041	15.2271	8.5354	13.0756	5.1164	30.4918	9.2168
μ	17.5763	16.1480	35.2587	303.6126	30.5256	54.1779	17.3004
σ	11.1148	8.2909	28.0450	749.3444	21.7045	29.7517	9.3080

Table 6.5: Progression analysis method comparison chart.

Acronyms: Evolutionary Optimization (EO), Trust Region (TR), Residual Spectrum Validation (RSV), Iterative Evolutionary Optimization (IEO), Trust Region using Band-Limited Differentiator Pre-Processing (TRD), Line model (Line), Non-Parametric Progression Analysis (NP).

Chapter 7

Fault Detection

7.1 Introduction

This chapter will focus on fault detection based on trend-based feature extraction methods. Anomaly detection methods are developed both for the parametric and the non-parametric features. The objective of these detection methods is to identify abnormal indicator behavior without a priori knowledge of specific fault signatures. Although a framework for fault recognition is suggested, diagnosis is given lower priority. This because there is not sufficient training data to cover all failure modes, thus making training and validation of such a classification system difficult. Further, it is from an operational point of view sufficient to perform a go / no-go decision and a crude fault localization. Should a component be suspected faulty, the aircraft will in any case be subject to a through manual inspection.

7.2 Classification

Most failure modes for most components are identifiable by fluctuations in the expected value or scatter level in one or more indicators associated with the component. Although different in architecture, both the parametric and the non-parametric feature extraction methods developed in the previous chapter extracts this information. For robust fault detection, it is however also necessary to perform a validation of each indicator step as well as direct threshold testing on certain critical indicators.

Edge occurrences should if possible be correlated with the aircraft maintenance log to verify that they really originate from equipment replacements. This generates the signal b_{fault} which return the step size for every step not occurring at the same moment as a maintenance action. For steps occurring

simultaneously with a maintenance action, b_{fault} remains zero.

Certain components, mainly rotors and engine shafts, have global unbalance thresholds which they not under any circumstance must supersede. To verify that the vibration levels for these components are within bounds, their unbalance indicators must be tested directly as a supplement to trend analysis. To avoid false alarms due to noisy data, this testing should however be done after the outlier component $\hat{s}$ has been removed.

This produces a total four metrics from each indicator at each point in time; a de-noised version of the indicator itself, fluctuations in scatter level, fluctuations in expected value, and unexpected step occurrences (Fig. 7.1). A component is however usually associated with several indicators. To identify the condition of a component it is necessary to evaluate the metrics from all the associated indicators. This provides the component state vector, consisting of $i - \hat{s}$, $a_{\hat{g}_w}$, $a_{\hat{c}}$ and b_{fault} for each indicator associated with the component, and describes the condition of the component at given instance in time.

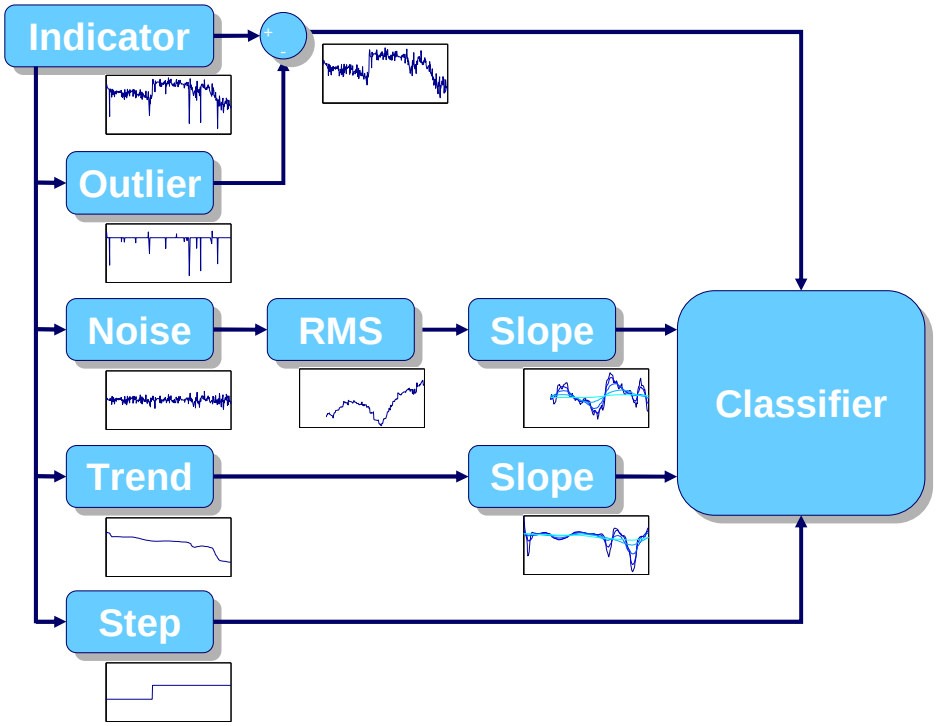

Figure 7.1: Indicator parametrization overview.

Using traditional HUMS methodology, a component is diagnosed by com-

paring the set of associated indicators to a baseline. This baseline must however be adapted to each aircraft, and is subject to change between maintenance actions. Using the component state vector, it is possible to use a global baseline which is not subject to change. This because the fluctuation metrics $a_{\hat{c}}$ and $a_{\hat{g}_w}$ are less sensitive to aircraft specific differences in the expected indicator value. Consequently, for indicators with large variation in expected value across aircraft, the baseline pays less attention to the absolute value, $i - \hat{s}$, and more attention to the fluctuation parameters $a_{\hat{c}}$ and $a_{\hat{g}_w}$. Inversely, for indicators with little variation between aircraft, such as shaft unbalance indicators, more significance is given to the indicator absolute value.

A simple fault detection method is to assign thresholds to each element in the component state vector. For the slope metrics, $a_{\hat{c}}$ and $a_{\hat{g}_w}$, both min and max thresholds should be used. For b_{fault} and $i-s$, it is sufficient to only apply max thresholds. A set of thresholds must be defined for each element in the state vector for each component on the aircraft. Once this is done, it will however be possible to apply the same thresholds to all aircraft of a given type.

Although such a method will permit detecting progressions deviating from normality, the method will not permit fault identification, i.e. diagnosis. To enable this, it is not only sufficient to detect that a progression is abnormal, it is also necessary to determine in which way the progression is abnormal. This can by done by a nonlinear mapping tool, such as a radial basis network.

All conditions has an expected value and an uncertainty for each element in the component state vector. This permits using the component state vector as the input to a radial basis network (Fig. 7.2). The inside of the radial basis network can be seen as a multidimensional space where the number of dimensions is equal to the number of inputs. In this space, each condition has a region, or cluster, at the position corresponding to the vector of expected values. The size of a region along each dimension is given by the uncertainty along the corresponding metric. For each input vector, the network will identify which region, and consequently which condition, the vector falls within. If a vector falls in the void between the clusters, it's interpreted as a unidentifiable anomaly.

Like with the threshold method, an instance of this network must be adapted to every component on the rotorcraft. Network calibration can be done either through expert knowledge or automated training. It is however difficult to obtain the necessary training data and expert knowledge to cover all conditions for every component. This due to the large number of failure modes to which a rotorcraft drive-train is susceptible, and due to the relatively low fault frequency on modern helicopters.

In order to perform a go / no-go decisions, it is however sufficient to be able to detect the normal state condition, for which sufficient training data exist for all components. Any component state vector not corresponding to normality must by definition be seen as abnormal. To maintain flight safety, abnormality detection is largely sufficient as any suspected anomaly will result in a manual inspection of the components in question. If a classifier is designed to only recognize deviation from normality, a threshold-based classifier is however preferred. This because the threshold-based approach is significantly less complex than a neural network.

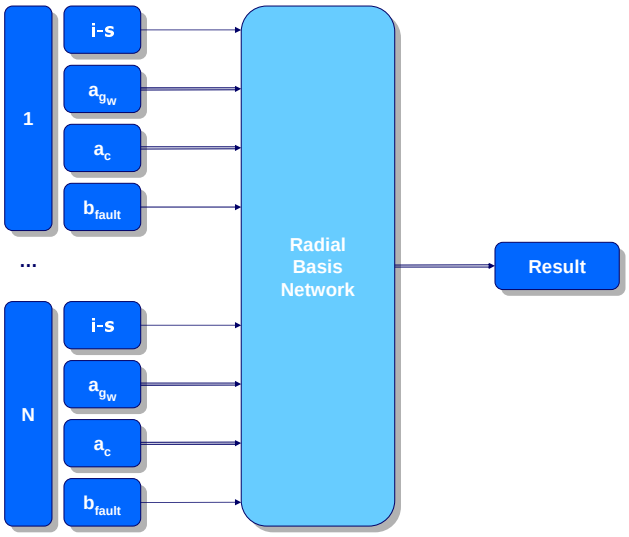

Figure 7.2: Component state classification.

7.3 Performance

The threshold-based method is tested using marked training-sets from AS332 L1 and L2 helicopters. The faulty sets are retrieved from clients' ground stations, isolating the propagation periods for documented defects. The healthy sets are data batches chosen on random outside the periods containing known defects, each case-number represents a different aircraft.

All the fault cases are loss of torque in the left ancillary gearbox intermediate gear fixing bolts. This allows the gear to assume a "wobbling" rotation patter, causing damage to its own tooth surface as well as the tooth surfaces of adjacent gears. When the gear rotates in an unbalanced manner, it forms a modulation between the shaft rotation frequency and the tooth meshing

tone. This creates modulation sidebands on each side of the meshing tone, with distance to the carrier equal to the shaft rotating speed. As unbalance increases, so does the modulation sideband energy, which is captured by the MOD indicator. Any fretting between the gear surfaces causes an increase in random noise, which increases the noise floor of the signal power spectrum. This phenomenon is captured by the RMSR indicator. Consequently, the indicators MOD and RMSR are chosen for detecting this fault type.

To detect the presence of faults, simple thresholds are applied to the expected value slope $a_{\hat{c}}$ and the scatter level slope $a_{\hat{g}_w}$ for each indicator. Scales 1 to 8 are chosen both for $a_{\hat{c}}$ and $a_{\hat{g}_w}$. Thresholds are based on the fluctuation-envelopes for cases 17 to 20. From these cases, the min and max values for each scale of $a_{\hat{c}}$ and $a_{\hat{g}_w}$ for each of the two indicators are retrieved, and used to form the threshold basis. The actual thresholds used for fault detection are the threshold basis multiplied by a factor. I.e. if the threshold factor is set to 120%, an alarm is generated whenever $a_{\hat{c}}$ or $a_{\hat{g}_w}$ is more than 120% above the max value or 120% below the min value experienced in the normal state training set.

Case 2 is illustrated with indicator decomposition, noise level estimate and slope shown in (Fig. 7.3, 7.4 and 7.5) for MOD and (Fig. 7.6, 7.7 and 7.8) for RMSR. The dotted lines are the threshold bases for each scale. Only scales 4 - 6 are plotted, in order to make the figures more readable.

A fault detection test on all the cases is conducted using threshold factors ranging from 90% to 150% in 10% steps, with results summarized in (Fig. 7.1). The "Length" column contains the length, in flight hours, of each data set. The "HUMS" column contains the ground-station detection results for each case, with the ground-station using traditional learned thresholds. Four of the fault cases (1, 3, 12 and 14) were missed by the ground-station diagnosis method, and were discovered by either metal chip-alarms or routine inspections. Case 7 was discovered by the operator manually inspecting the indicators and signals. It can thus not be known if the HUMS would have generated an alarm, had it not been detected manually.

It is not known if, and how many, false alarms were generated by the healthy state datasets. As a global average, a HUMS generates alarms in the magnitude of 4 to 12 per 1000 flight hours. With the component in question being one of about 80 components monitored on the AS332, it is likely to believe that it would produce a proportional number of false alarms.

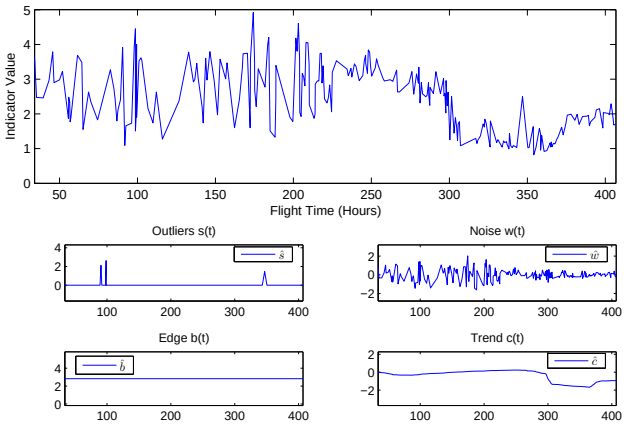

Figure 7.3: Case 2 MOD raw and decomposed.

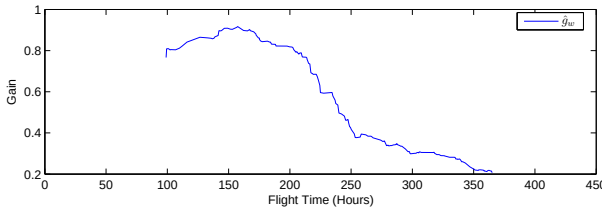

Figure 7.4: Case 2 MOD scatter energy.

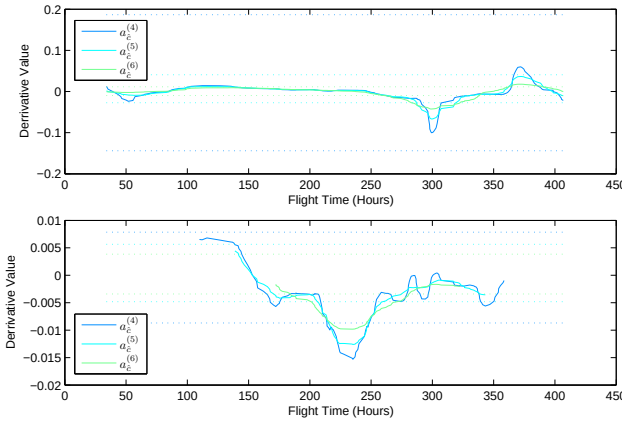

Figure 7.5: Case 2 MOD expected value and scatter energy slopes.

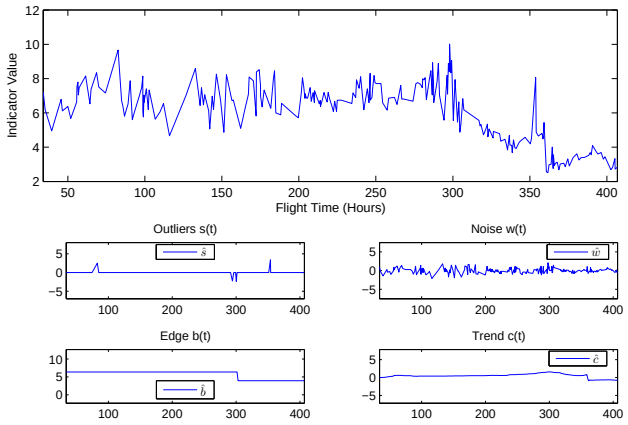

Figure 7.6: Case 2 RMSR raw and decomposed.

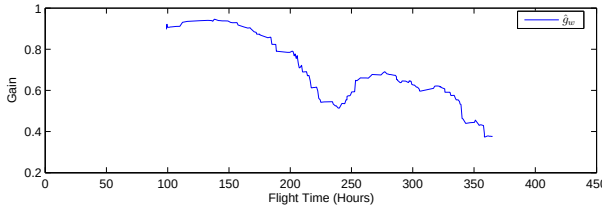

Figure 7.7: Case 2 RMSR scatter energy.

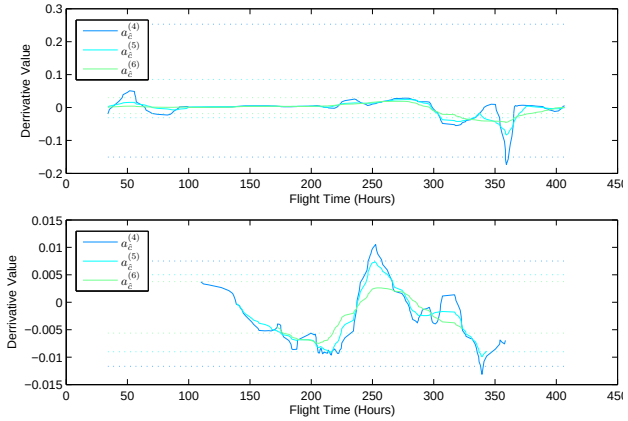

Figure 7.8: Case 2 RMSR expected value and scatter energy slopes.

Case	State	Length	HUMS	90	100	110	120	130	140	150
1	Faulty	86.03	no	yes	yes	yes	yes	yes	yes	yes
2	Faulty	372.8	yes	yes	yes	yes	yes	yes	yes	yes
3	Faulty	336.92	no	yes	yes	yes	yes	yes	yes	yes
4	Faulty	78.52	yes	yes	yes	yes	yes	yes	yes	yes
5	Faulty	267.37	yes	**no**	**no**	**no**	**no**	**no**	**no**	**no**
6	Faulty	50.92	yes	yes	yes	yes	yes	yes	yes	yes
7	Faulty	194.47	n / a	yes	yes	yes	yes	yes	yes	yes
8	Faulty	251.02	yes	yes	yes	yes	yes	yes	yes	yes
9	Faulty	336.88	yes	yes	yes	yes	yes	yes	yes	yes
10	Faulty	232.45	yes	yes	yes	yes	yes	yes	yes	yes
11	Faulty	77.43	yes	yes	yes	yes	yes	yes	**no**	**no**
12	Faulty	49.44	no	yes	yes	yes	yes	yes	yes	**no**
13	Faulty	195.99	yes	yes	yes	yes	yes	yes	yes	yes
14	Faulty	36.53	no	yes	yes	yes	yes	yes	yes	yes
15	Healthy	877.78	n / a	no	no	no	no	no	no	no
16	Healthy	2019.85	n / a	no	no	no	no	no	no	no
17	Healthy	1608.58	n / a	**yes**	no	no	no	no	no	no
18	Healthy	1479.9	n / a	**yes**	no	no	no	no	no	no
19	Healthy	1093.09	n / a	**yes**	no	no	no	no	no	no
20	Healthy	218.5	n / a	no	no	no	no	no	no	no
21	Healthy	148.74	n / a	no	no	no	no	no	no	no

Table 7.1: Authentic test cases.

Using a 90% threshold factor obviously causes false alarms in the training sets, cases 17 - 20. The reason that the other healthy states sets are not producing any false alarms is that they have significantly lower fluctuation levels than the training sets. Threshold factors of 100% to 130% provides good results for the data at hand, although a factor of 100% is unadvisable for training data with more representative fluctuation levels. Case 5 is the only non-detection for this range of threshold factors, and requires a factor of 60% to be detected, which obviously will create an unacceptable false alarm rate. Indicator decomposition, noise level estimate and slope is shown in (Fig. 7.9, 7.10 and 7.11) for MOD and (Fig. 7.12, 7.13 and 7.14) for RMSR.

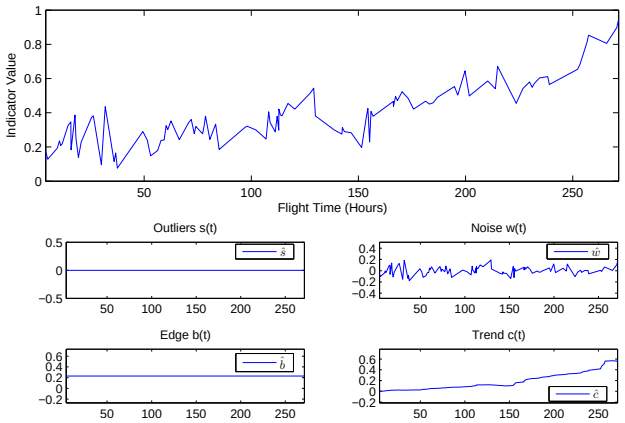

Figure 7.9: Case 5 MOD raw and decomposed.

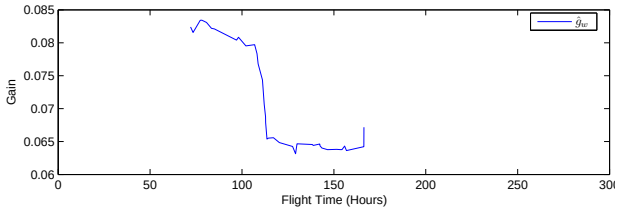

Figure 7.10: Case 5 MOD scatter energy.

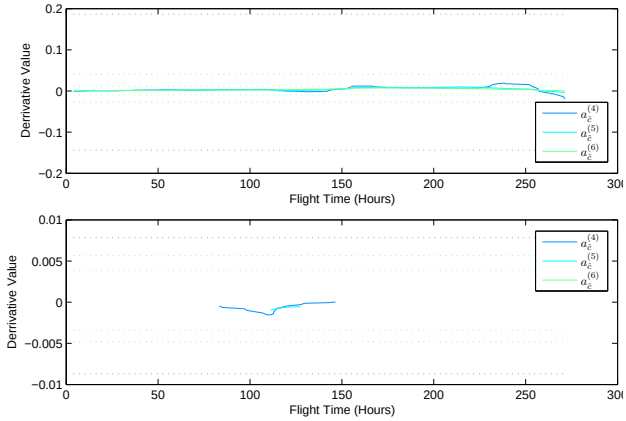

Figure 7.11: Case 5 MOD expected value and scatter energy slopes.

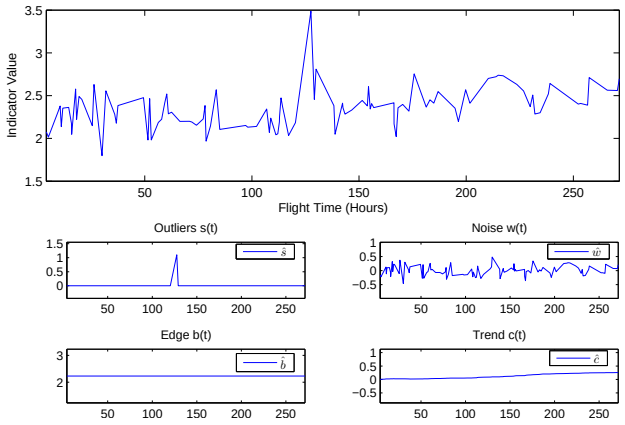

Figure 7.12: Case 5 RMSR raw and decomposed.

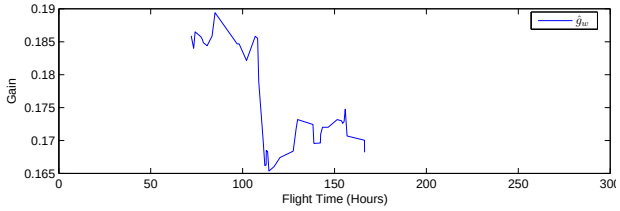

Figure 7.13: Case 5 RMSR scatter energy.

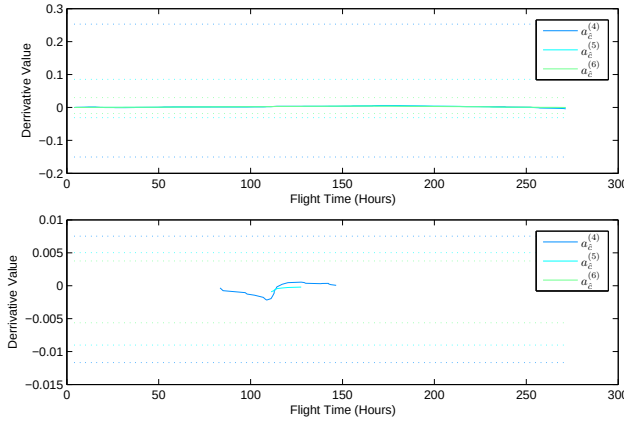

Figure 7.14: Case 5 RMSR expected value and scatter energy slopes.

Studying the raw indicator plots, trends are indeed visible. The detection failure is however due to the exceptionally low magnitude for both indicators in this case. Even though trends are clearly visible to a human analyst, the slopes, i.e. augmentation or diminution per hour, is still small due to the abnormally low magnitude of both indicators. A way to circumvent this problem is to not evaluate the slopes directly, but rather to normalize slopes by scatter level $\hat{g}_w$. This will in fact mimic the human approach to finding trends in noisy data; to identify patterns which clearly rise clearly above the random scatter. Using the test data at hand, this is however not possible. Due to the short duration of some of the fault cases, the scatter energy $\hat{g}_w$ can not be estimated, thus precluding the use of this method.

7.4 Conclusion

This chapter has been more concerned with fault detection than diagnosis and prognosis. The reason for this being that there is little authentic fault data on record, making it difficult to train and validate a fault recognition system. The only data existing in rich supply is normal state data. Consequently, it is easier to create a system capable of recognizing deviation from normality, and validating this on the few fault case datasets that do exist. From an operational point of view this is largely sufficient, as any suspected damage will in any case result in manual inspection of the aircraft.

Using the feature extraction methods developed in the previous chapter with training data representing a healthy condition of an asset, it is possible to detect most anomalies without aircraft specific configuration of the fault detection algorithm. Further, this method has a better detection rate than traditional threshold based methods. Although testing only a single failure mode is insufficient to determine if the method has potential for other fault types, the method still show promise. A strong point of the method is that it does not require any sort of configuration or training by the user. This is a major advantage compared to traditional methods, which are prone to miss-learning by the user, leaving the system with reduced fault detection capabilities.

Chapter 8

Conclusion

8.1 General

Fault detection in helicopter engines, rotors and transmission systems has been an area of intense research for over one and a half decade. Even with this continuous development into HUMS, as well as other areas of accident prevention, rotorcraft remains overrepresented in the accident statistics compared to equal size airplane. Consequently, there is still work to be done within the HUMS domain as well as other safety enhancing technologies.

Another aspect of HUMS is maintenance cost reduction. Military operators have already started to adapted maintenance work to benefit from the fault detection capabilities of HUMS, retiring components "on condition" [23]. Although civilian aviation is somewhat behind on this, due to rigid regulatory bodies, this is also an important are of research for the aircraft manufacturers. Using military maintenance practice as a proof of concept, it is likely that civil aviation will follow this trend implementing maintenance credit for components which can reliably be monitored by HUMS.

8.2 User Friendliness

An aspect easily neglected when working with complex reasoning systems is the user interface. Although the robustness and accuracy of the reasoning algorithms if of vital importance for any decision aid system, user interface is of equal importance in order to establish user confidence. In the context of HUMS, a somewhat neglected area has been data mining and management. This mainly due to the fact that all the data analysis needs, both for operator and OEM, were not clear when the functional specifications for HUMS were developed. Consequently, HUMS software was not equipped with the data

analysis tools necessary to satisfy the needs of all clients, and no mechanism were included to share HUMS data with third party tools when necessary.

These shortcomings have been addressed by the data structuring work detailed in the chapter on data migration, and facilitate the use of third party tools to perform more in-depth analysis of HUMS data. This work also addresses the problem of data availability for the OEM researchers and support personnel, as well as reducing client workload associated with data sharing. Further, this study has been a contributer to a project facilitating discrepancy reporting between client and OEM, significantly reducing client workload associated with discrepancy reporting as well as cutting response time for the OEM. This latter point probably being an example to follow for support services dealing with other aspects of the aircraft than HUMS.

8.3 Reliability

The main focus for this study has been on fault detection, which has aimed at increasing system reliability and reducing operator workload. In the utmost consequence it can be said that traditional threshold based methods are capable of detecting all faults, provided that all thresholds are correctly adjusted. Experience does however show that indicator thresholds are not aways correctly set. This due to the fact that the correct tuning of any threshold changes as a function of outside factors like maintenance interventions. In order to cope with this reality, it has been necessary to explore new concepts in order to reach the aim for this study; a reliable fault detection method not in need of any user configuration.

Using progression analysis as feature extraction for faults not easily detectable through traditional means has shown increased detection rate and reduced false alarm rates compared to traditional threshold based methods. Although the test data available is too scarce to generalize, progression analysis shows promise as an alternative to conventional methods. Another advantage of progression analysis is that it does not require any configuration by the operator. By combining threshold based detection methods where applicable with progression analysis for components not easily monitored by traditional means, it is possible to create an autonomous HUMS. Such a system has clear safety benefits as it is not susceptible to miss-configuration by the user. This is probably the most interesting aspect of this study, as there are currently no systems of the marked capable of functions autonomously. Together with the increased detection rate demonstrated in the case study, these advances deliver the technology necessary to produce a commercial solution well ahead of the competing systems.

8.4 Forward Perspectives

Although this study has produced original and promising results, much work remains in validating and industrializing these methods. This mainly due to the fact that insufficient validation data exist on record. Significant work on data migration was done as a direct response to this problem, but the resulting data handling infrastructure came in place too late for this study to take significant advantage of it. Still, this infrastructure remains in place for other studies, current and future, within the HUMS domain.

An area in need of further study is the correlation between vibration signatures and environmental factors. Although a framework de-correlation of vibration signatures and environmental factors were developed as a part of this study, there is still work to be done to gain an understanding of how and why the environment affects the vibration signature of a transmission system. Using the latest generation airborne segment, which entered service at the end of this study, gains access to the full range of contextual parameters recorded on the aircraft, and makes it possible to correlate vibration signatures against a wide range of environmental factors. By gathering maintenance logs from the operator it will also be possible to investigate any connection between erroneous equipment installation and vibration signatures' sensitivity to environmental factors.

Another research area is validation and industrialization of the progression analysis methods. Although these have been found effective on a limited number of test cases, it is still to test the effectiveness of these methods on a wider variety of failure modes. Using data from the above discussed data warehouse will facilitate testing against all confirmed fault cases on record. This is already part of an ongoing study aiming both at including methods from this study as well as testing new methods [46] to select the set of methods which will be deployed on the next generation HUMS ground station.

Looking at HUMS from a long term perspective raises the question of maintenance credit, like extended TBO as a consequence of reliable condition monitoring. If the effectiveness of existing fault detection methods can be documented, it will be possible to change aircraft maintenance procedures so that both manual inspection and premature component retirement is reduced. Deploying this at a large scale will have a profound impact on rotorcraft maintenance cost, and life cycle cost in total. A study is already launched aiming at identifying the areas where credit can be obtained, as well as trying to establish a framework for documenting and certifying automated monitoring techniques which can be accepted by the civil regulatory bodies. This will open the door for significant reduction in aircraft maintenance cost and downtime, while at the same time increasing safety.

Appendix A

Mathematical Notations

A.1 Moving Median

Moving median filters are frequently used for outlier removal in signal processing. The moving median works similar to FIR or IIR filters, but extracts the median rather than the sum. Median filters can have individual weight for each tap, but this is not exploited here.

$$mm(x, n, K) = median_{k=n-K}^{n}[x(k)] \tag{A.1}$$

A.2 Windowed RMS

A windowed RMS is used for estimating signal energy at a limited segment of the input signal. A windowed RMS with window size K of a signal with length N gives an output of length $N - K$.

$$wrms(x, n, K) = \frac{1}{K} \sum_{k=n-K}^{n} (x(k) - \bar{x})^2 \tag{A.2}$$

A.3 Wavelets

The wavelet theory discussed here is intentionally cut short. For better understanding of wavelet theory and applications, the reader should refer to the relevant literature, such as [27], [19] and [8].

A.3.1 Continuous Wavelet Transform

Wavelet theory shows that a signal $f(t)$ can be represented as a weighted sum of functions $\psi_{j,k}(t)$ (Eq. A.3). The integer indices j and k represents scaling and translation of some arbitrary wavelet base function $\psi(t)$ (Eq. A.4). The better choice of $\psi(t)$ depends on the application. For de-noising and compression, a wavelet is chosen which confines noise and signal in different parts of the $d_{j,k}$ coefficient matrix (Eq. A.5). For event detection, a wavelet is chosen which gives the event an easily recognizable signature.

$$f(t) = \sum_{j,k} d_{j,k}\psi_{j,k}(t) \tag{A.3}$$

$$\psi_{j,k}(t) = 2^{j/2}\psi(2^j t - k) \tag{A.4}$$

$$d_{j,k} =< f(t), \psi_{j,k}(t) > \tag{A.5}$$

The above definitions will for a finite length signal give a coefficient matrix with finite length in the translation (k) dimension. The number of possible scalings (j) is however infinite. This means that reconstructions is not possible unless coefficients for an infinite number of scales are computed.

A.3.2 Discrete Wavelet Transform

The Discrete Wavelet Transform (DWT) overcomes this by expanding over only a limited number of scales, leaving the remaining part of $f(t)$ in an approximate vector a_k. This way the detail matrix $d_{j,k}$ and the approximate vector a_k will together hold all information necessary to reconstruct the original signal (Eq. A.6). The number of scales must be chosen so that some intelligent partition of signal features is obtained between the approximate vector and the different scale vectors of the detail matrix.

$$f(t) = \sum_{k} a_k\phi(t-k) + \sum_{j,k} d_{j,k}\psi_{j,k}(t) \tag{A.6}$$

$$< \psi(t), \phi(t) >= 0 \tag{A.7}$$

The wavelet function $\psi(t)$ and the scaling function $\phi(t)$, used for extracting the approximate, must be orthogonal (Eq. Eq. A.7). Consequently, the DWT can only be calculated using wavelet base functions for which an orthogonal scaling function exist.

Practical implementations are normally performed using a dyadic filter bank, where the filters are derived from the wavelet and scaling functions (Eq. A.8 and A.9).

$$\phi(t) = \sum_n h(n)\sqrt{2}\phi(2t-n) \tag{A.8}$$

$$\psi(t) = \sum_n h_1(n)\sqrt{2}\phi(2t-n) \tag{A.9}$$

This forms a recursive equation set where the initial approximate vector is the transform input $a_{j_0} = f(t)$ (Eq. A.10 and A.11). The input to any scale a_j is the approximate output from the previous one.

$$d_{j,k} = \sum_n h_1(n-2k)a_{j+1,n} \tag{A.10}$$

$$a_{j,k} = \sum_n h(n-2k)a_{j+1,n} \tag{A.11}$$

Reconstruction (Eq. A.12) starts with the lowest level detail and approximate vectors, which are used for reconstructing the second lowest approximate. The algorithm the recurses back until the original a_{j_0} coefficients are obtained. Consequently, only the lowest level approximate vector, in addition to the entire detail matrix, is necessary for reconstruction of the original input.

$$a_{j+1,k} = \sum_n h(k-2n)a_{j,n} + \sum_n h_1(k-2n)d_{j,n} \tag{A.12}$$

The $2k$ translation factor of the filters stems from the factor-of-two relation between each scale (Eq. A.4). This ensures that each scale has half the number of coefficients of the previous one. Consequently, the total number of coefficients for any number of iterations will be equal to the number of input samples. By guarding only high-value coefficients, a reasonably accurate reconstruction can be made using only a fraction of the original coefficients. By disposing of the coefficients representing noise, reconstruction gives a de-noised version of the original input.

A.3.3 Stationary Wavelet Transform

For de-noising, better results are often obtained using the Stationary Wavelet Transform (SWT) [8]. SWT differs from DWT by upsampling the filters at

each scale instead of downsampling the coefficients. This creates a redundancy as each layer (j) produces the same number of coefficients as the input signal. The tradeoff is better time (k) precision for coarse scales.

This study deals only with finite signals. In order to keep filter output length the same as input length, while reducing transients, signals / coefficients are padded at start and end. Padding consist of samples having the mean value of the K first / last samples. This provides better results than periodic padding (circular convolution), as the start and end of the signals used in this study can have very different amplitude.

A.4 Nonlinear Optimization

Optimization problems can normally be transformed into minimization problems, which consist in finding the x which minimizes an object function $f(x)$. The variable x can be a vector, so that $f(x)$ is a function of one or more variables. For linear functions, such as polynomials, the solution can be found using symbolic derivation. For non-linear problems, the solution must be found using other methods.

Problems, for which a point derivative can be calculated, can be solved using steepest descent methods. More general methods are Direct Search and Evolutionary Optimization. These methods do not require the object function to be point derivable, nor even continuous.

A.4.1 Trust Region

Trust Region [34] is a frequently used steepest descent algorithm. The algorithm approximates the object function $f(x)$ by a trivial function $d(x)$, normally the first few terms of a Taylor series, which provides a reasonably accurate approximation within a region, the trust region, surrounding the initial current position. Once the position minimizing $d(x)$ has been found, this point is accepted as the new current position, provided that it provides lower output from $f(x)$ than the previous position. Each time this criteria is meet, the trust region is expanded. Upon failure, i.e. the previous position outperforms the new one, the current position is kept and the trust region contracted.

1. Approximate $f(x)$ by trivial function $d(x)$ in region R around x_0
2. Find the value for x, x_1, which minimizes $d(x)$
3. If $f(x_1) < f(x_0)$ then set $x_0 = x_1$, expand R and goto 1

4. Else contract R and goto 1

The process iterates until some stopping criterion is meet. This will normally be the step length converging to zero, max number of iterations, or max execution time. All examples in this study uses a step length of less than 10^{-8} as stopping criterion.

A.4.2 Evolutionary Optimization

Evolutionary algorithms [15] exist in several variants. This study makes use of a method which does not require discretization of the parameters, and uses the same number of individuals in each generation for a limited number of generations. The algorithm is initiated by creating a population of parameter combinations which are chosen on random from an initial range of possible values. Once performance is tested for all individuals, the best individuals, i.e. the best combinations of parameters, are labeled elite individuals and transferred to the next generation. The other individuals for the next generation are generated from the remaining population of the current generation by either crossing or mutating. Crossing means that two children are created from two parents by choosing each parameter for each child from one of the parents. Which parameter is chosen from which parent is determined by a binary random process. The probability that an individual is chosen as a parent is proportional with its rank among the individuals. Consequently, the best individuals will normally be chosen as parents several times within the same generation. Finally, a set of individuals are mutated. This means that they are randomly displaced in parameter space, driven by a Gaussian random process. Once the parameters for the next generation are ready, the performance of the next generation is calculated. This process iterates until a performance goal or the maximum number of generations is reached.

Mutation is necessary in order to create new parameter values, as crossing simply generates new combinations of existing values. The Gaussian variance along each dimension is for the first generation set as the span of the initial range. The variances for the following generations are set as a function of initial variance and generation number, so that the standard deviation reaches to zero when max generation count is reached. This way, large parts of the parameter space is explored early in the evolution, before the individuals converges on the optimal solution.

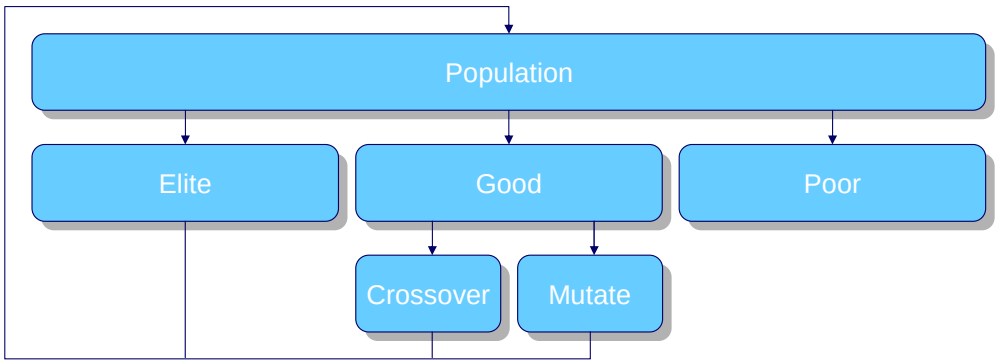

Figure A.1: Evolutionary optimization overview.

A.5 Classification Systems

Classification systems are system for categorizing some input. A classification system accepts an input vector, generating an output vector. The output usually corresponds to class probabilities, so that the value of an output vector element is equivalent to the probability of the input vector belonging to the associated class.

In the most general of terms, a classification system can be seen as a mapping tool, mapping one vector to another. Simple classification systems can be based on polynomials or Gaussian functions. More complex systems, such as artificial neural networks and fuzzy logic, are capable of more complex non-linear mapping. Some classification systems can be estimated, trained, through a set of marked training sets. I.e. input vectors where the output vector is known. Other classification systems are best designed "by hand" using expert knowledge.

A commonly used non-linear mapping tool is the radial basis network. A radial basis layer, left part of figure A.2 [6], consist of a number of neurons A, each with an R dimensional position in feature space. It accepts an R element feature vector as input, r, and computes the euclidean distance between the vector and each neuron. The A dimensional output, which reflects the distance to each neuron, is multiplied with a bias and sent through a radial basis function. The radial basis function, of form $f(x) = e^{-x^2}$, returns 1 for $x = 0$ and converges quickly to 0 for input larger than 0. Consequently, the layer outputs 1 for neurons with coordinates matching the input vector, and a value between 1 and 0 for neurons further away. The rate of descent for is controlled by the bias. I.e. the bias controls the active radius, the area for which a neuron produces a significant output.

The active region of a neuron is a multidimensional sphere in input space. A class corresponds to region in input space, covered by the active region of one or more neurons. As some classes are covered by more than one neuron, the radial basis layer is often followed by a linear layer. The linear layer simply fuses together the outputs from the neurons corresponding to the same class, so that the number of network outputs corresponds to the number of classes.

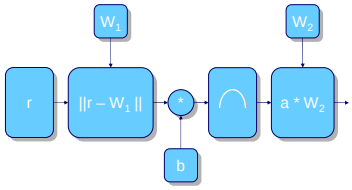

Figure A.2: Radial basis network.

Two training strategies exist for radial basis networks. One is to fix the bias and create one neuron for each training vector. This way, the network is sure to classify all the training vectors correctly. An alternative approach is to limit the number of neurons, and adjust neuron positions and biases so that the total miss-classification of the training set is minimized. The latter approach will generate a network with fewer coefficients / neurons, and in some cases better generality.

Appendix B

IT Notations

B.1 Databases

Enterprise business solutions typically generates complex data storage needs, such as centralized storage for multiple users involving simultaneous multi users reading and writing. This creates problem that goes far beyond the scope of the file locking functions provided by most file systems. Standard Query Language (SQL) databases are solutions that are capable of providing such functions.

An SQL database is in all simplicity a patch of shared memory. SQL is the languages used for reading and writing to this memory. Commands in SQL format are sent from a user to the database engines, which return the result (if any) to the user. The mode of communication is specific to each database engine. Most database engines are background processes accessible through TCP/IP and / or platform specific data pipe systems. Database engines accessible through TCP/IP are typically accessible over network, and provides client connectivity implementations for several platforms. A client connectivity API is necessary even for TCP/IP based database engines, because there exist no common application layer protocol for SQL database (the layer between TCP and SQL).

The data in an SQL database is organized in tables. There exist SQL commands for inserting, retrieving, updating and deleting table data, as well as crating, modifying and deleting tables. Most database engines are capable of serving multiple clients at the same time, and provides rigorous mechanisms for ensuring data consistency in multi user environments. Enterprise scale database engines also provide sophisticated data mining functions and integrated backup solutions.

In scenarios involving multi user data access and data trafficking over

networks, developers can save substantial time by using third party database servers rather than developing their own proprietary solutions. Further, using open standards like SQL makes solutions more scalable and more easily accessible through other third party systems.

B.2 Object Oriented Programming

Object oriented programming has received increasing attention over the last decade, and has become the defacto standard for complex software projects. Object oriented programming means in essence nesting functions that belong together into objects. Some object oriented languages also deploys packages or namespaces, for nesting classes that belong together.

An illustrative problem example is file i/o operations. Using classical C-style function calls, file opening, reading, writing and closing are managed by a set of global functions. All of these functions use a file handle as input or output, where the handle is a reference to a place in memory where the file is stored. An object oriented approach to the same problem is creating an object containing the same functions, and with the file handel being part of the internal memory of object.

The object oriented approach makes code development tidier, as there is no need to find globally unique names for each function. When designing an object to manage a certain task, an important aim is usually to make the external interface to the object as simple as possible, thus concealing the complex inner working of the task. This allows programmers to develop tidy code libraries which are highly reusable.

B.2.1 Interface Programming

The externally accessible functions and variables of an object is referred to as the object interface. An essential part of object oriented programming is the possibility for objects to implement pre-defined interfaces. If an object announces that it is implementing an interface, it means that is providing an implementation for all the methods and variables defined in the interface.

A typical example of interface programming is database access. All SQL database engines typically provide the same functionality, but has different access protocols. Thus, object models like Java, ActiveX and .net provide a hierarchy of interfaces representing the standard database objects such as connection, command and result set. The database providers then provide implementations for their specific protocols using these standard interfaces. Consequently, a client application can relate only to the standard

interfaces, without any knowledge about the underlying database specific protocols. This means that the same client code can interact with several different database engines by simply loading the interface implementations corresponding to the desired database engine dynamically at runtime.

Interface programming is a powerful tool allowing application functionality to be split in layers, where each layer can have multiple implementations. Reusable objects such as database drivers and multimedia codecs can then be shared between applications. Further, a given application can seamlessly switch between several implementations of a given interface hierarchy, for instance to support multiple video formats, without any specific code to accommodate each implementation.

B.2.2 Java

Java is a high level programming language developed by Sun Microsystems. A Java application does not run directly on the operating system, but on a Java Virtual Machine (VM). The VM is a kind of operating system inside the operating system, and exists for most operating systems and hardware platforms. Thus, the VM acts as an abstraction layer, making the underlying platform transparent. This makes Java applications highly portable, as the same compiled code will run on almost any platform.

The VM is responsible for resource allocation, including memory allocation and deallocation. Object instance memory is typically allocated on the heap, allowing methods to return object instances by reference. The VM is also responsible for counting the references to each object instance. Once the reference count to an object drops to zero, the object instance memory is automatically released.

Java objects are organized in packages, with each package containing objects that logically belong together. A package can also contain sub-packages, forming an hierarchical system. A Java application can effortlessly load objects at runtime, either from a predefined library location, or from an explicitly defined source. Thus, machine code can be imported into the running assembly from any source, remote or local, or even from code compiled at runtime.

B.2.3 Component Object Model

A Microsoft Dynamic Link Library (DLL) is a file containing executable code but which has no default entry point. Accessed to a DLL from an executing assembly (EXE or DLL) is obtained by loading the DLL into memory and specifying the address inside the DLL where to start execution. Once the

DLL instruction path ends, control is returned to the calling assembly. This corresponds to a C-style function call. A DLL can be loaded when the executing assembly is loaded, static linking, or when the executing assembly calls one of the functions in the DLL, dynamic linking.

A library model which only allows for sharing global C-style functions is an obvious disadvantage in modern object oriented programming. This problem is solved by the Component Object Model (COM). A COM object is a standardized DLL containing four functions. Three of these are for component registration and memory management. The fourth returns an instance of the Class Factory. The Class Factory is an object which manages the objects inside the COM object. Given a unique numeric reference to an object implementation, CLSID, the Class Factory will return an instance of the requested object. Thus, COM is the means to access shared C++ style classes using C-style function calls.

Any COM object is registered in the system registry with a globally unique CLSID, a human readable ProgID, and a reference to the DLL hosting it. An executing assembly uses a system call to load the object corresponding to a given CLSID. The operating system answers this call by loading the DLL specified in the CLSID registry entry, and asking the DLL Class Factory for an instance of the object corresponding to the CLSID.

COM simplifies memory management by making memory allocation the responsibility of the callee, rather than the caller, which is is the case for C. Although making programming easier, this has a certain performance cost as object instance memory is allocated on the heap rather than the stack. A particularity of COM, compared to Java and .net, is that garbage collection is not managed by the operating system. Consequently, each object must provide its own reference counting and memory deallocation code. This is managed by methods defined in a Microsoft defined interface, IUnknown, which all COM objects must implement.

As any object can support multiple interfaces, an object-level type cast mechanism is provided through IUnknown. This interface contains a method for fetching a pointer to an interface corresponding to a given interface ID, IID. Microsoft has defined a wide range of interfaces ranging from window controls to sound compression codecs. One essential interface is IDispatch. The IDispatch interface defines a method accepting an array of strings and pointers as input. This allows for providing an array containing the name of a method, as well the arguments for the method. IDispatch will process the request by calling the C++ method corresponding to the specified name, using the specified argument values.

This mechanism allows an assembly to call the methods of any COM object implementing IDispatch without needing to be compiled to run any

other interface than IDispatch. Consequently, an assembly can generate code "on the fly", and feed it to an object instance through its IDispatch interface. This functionality is deployed by most win32 based script engines, including Microsoft Internet Explorer and Matlab. COM objects implementing IDispatch are commonly referred to as automation or ActiveX objects.

B.2.4 .net

This technology is developed by Microsoft, and is a continuation of the COM / ActiveX concept. A .net application runs inside a framework, much like a Java application. Consequently, memory deallocation is no longer the responsibility of the object programmer, but is handled by the framework. Framework implementations does however only exist for Microsoft operating systems, making .net applications non-portable to other platforms. This decrease in generality, compared to Java, gives .net applications access to Microsoft specific features such as the system registry and the real-time library DirectX. Objects developed in .net are cross compatible with COM, meaning that they can both make use of existing COM objects as well as exposing their own COM interface.

Objects developed in .net are organized in namespaces, which serve the same function as Java packages. A .net application can effortlessly load objects at runtime, either from a predefined library location, or from an explicitly defined source. Thus, machine code can be imported into the running assembly from any source, remote or local, or even from code compiled at runtime.

List of Figures

List of Tables

Bibliography

[1] CIVIL AVIATION AUTHORITY. Cap693, acceptable means of compliance, helicopter health monitoring, caa aad 001-05-99, 1999.

[2] Lionel Stuart Barry. Aircraft health and usage monitoring system. Patent EP0407179A1, 1991.

[3] Eric Bechhoefer and P.F. Bernhard Andreas. Hums optimal weighting of condition indicators to determine the health of a component. In *American Helicopter Society 60th Annual Forum*, Baltimore, USA, 2004.

[4] J.S. Bendat and A.G. Piersol. *Random Data; Analysis and Measurement Procedures, Second ed.* Wiley, New York, 1986.

[5] S. Charbonnier, C. Garcia-Beltan, C. Cadet, and S. Gentil. Trends extraction and analysis for complex system monitoring and decision support. *Engineering Applications of Artificial Intelligence*, 18:21–36, February 2005.

[6] S. Chen, C. F. N. Cowan, and P. M. Grant. *Orthogonal Least Squares Learning Algorithm for Radial Basis Function Networks*, volume 2. IEEE Transactions on Neural Networks, 1991.

[7] R.L. Claypoole, R.G. Baraniuk, and R.D. Nowak. Adaptive wavelet transforms via lifting. In *IEEE Conference on Acoustics, Speech, and Signal Processing*, 1998.

[8] R. R. Coifman and D. L. Donoho. Translation-invariant de-noising. Technical report, Stanford University, Department of Statistics, 1995. To Appear, Wavelets and Statistics, Anestis Antoniadis, ed. Springer-Verlag Lecture Notes.

[9] J. Colomer, J. Melendez, and F. Gamero. Qualitative representation of process trends for situation assessment based on cases. In *IFAC World Congress*, Barcelona, Spain, 2002.

[10] H.J. Decker and D.G. Lewick. Spiral bevel pinion crack detection in a helicopter gearbox. In *American Helicopter Society's 59th Annual Forum*, Phoenix, Arizona, USA, 2003.

[11] Joint European Helicopter Operators Committee (EHOC) / Association Européenne des Constructeurs de Matériel Aérospatial (AECMA) / Aerospace Industries Association (AIA) HUMS task force. Hums task force report, 2002.

[12] David L. Donoho. De-noising by soft-thresholding. *IEEE Transactions on Information Theory*, 41(3):613–627, 1995.

[13] M.A. Essawy, S. Diwakar, S. Zein-Sabatto, and A.K. Garga. Fault diagnosis of helicopter gearboxes using neurofuzzy techniques. In *52nd Meeting of the Society for Machinery Failure Prevention Technology (MFPT)*, 1998.

[14] W. Gersch, T. Brotherton, and S. Braun. Nearest neighbor-time series analysis classification of faults in rotating machinery. *Journal of Vibration, Acoustics, Stress, and Reliability in Design*, 105:178Ű–184, 1983.

[15] David Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Kluwer Academic Publishers, Boston, MA, 1989.

[16] T. Harris and A. Kohonen. S.o.m. based machine health monitoring system which enables diagnosis of faults not seen in the training set. In *The 1993 International Joint Conference on Neural Networks*, 1993.

[17] J.J. Zakrajsek H.J. Decker, R.F. Handschuh. An enhancement to the na4 gear vibration diagnostic parameter, technical report nasa tm-106553, arl-tr-389. Technical report, NASA and the US Army Research Laboratory, 1994.

[18] A.J. Hoffman and N.T. van der Merwe. The application of neural networks to vibrational diagnostics for multiple fault conditions. *Computer Standards & Interfaces*, 24:139Ű–149, 2002.

[19] I. Daubechies. *Ten Lectures on Wavelets*. SIAM Publications, 1992.

[20] G. Kitagawa and W. Gersch. A smoothness prior time-varying ar coefficient modeling of nonstationary covariance time series. *IEEE Transactions on Automatic Control*, 30:48Ű–56, 1985.

[21] Renata Klein. Method and system for diagnostics and prognostics of a mechanical system. Patent WO2004059399A2, 2004.

[22] Brian D. Larder, R. Callen, and R. Salter. Intelligent management of hums data: Development of a vibration health monitoring anomaly detection system. In *American Helicopter Society 62th Annual Forum*, 2006.

[23] W. Lewis, C. Perry, and J. Keller. Airworthiness releases as a result of condition based maintenance. In *32nd European Rotorcraft Forum*, Maastricht, Nederland, September 2006.

[24] J.E. Lopez, I.A. Farber-Yeldman, K.A. Oliver, and M.W. Protz. Hierarchical neural networks for improved fault detection using multiple sensors. In *American Helicopter Society 52nd Annual Forum*, 1996.

[25] P.-J. Lu, M.-C. Zhang, T.-C. Hsu, and J. Zhang. An evaluation of engine faults diagnostics using artificial neural networks. *Journal of Engineering for Gas Turbines and Power*, 123, 2001.

[26] J. Ma. Energy operator and other demodulation approaches to gear defect detection. In *Proceedings of the 49th Meeting of the Society for Machinery Failure Prevention Technology (MFPT)*, Virginia Beach, Virginia, USA, 1995.

[27] S. Mallat. A theory of multiresolution signal decomposition: the wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11:674–693, 1989.

[28] H.R. Martin. Statistical moment analysis as a means of surface damage detection. In *Seventh International Modal Analysis Conference*, Schenectady, New York, USA, 1989.

[29] Gary M. McBrien. Vibration monitoring system for gas turbine engines. Patent WO03056284A2, 2003.

[30] P.D. McFadden. Examination of a technique for the early detection of failure in gears by signal processing of the time domain average of the meshing vibration. *Mechanical Systems and Signal Processing*, 1(2):173Ű–183, 1987.

[31] P.D. McFadden. A technique for calucalting the time domain averages of the vibration of the individual planet gears and sun gear in an epicyclic gearbox. *Journal of Sound and Vibration*, 144(1):163Ű–172, 1991.

[32] P.D. McFadden and I.M Howard. The detection of seeded faults in an epicyclic gearbox by signal averaging of the vibration. Technical report, Australian Defense Department: Defense Science and Technology Organization Aeronautical Research Laboratory, ARL-PROP-R-183, 1990.

[33] E. Mermoz. *Automatisation du Diagnostic Vibratoire des Transmissions de Puissance d'Helicoptere*. PhD thesis, Ecole Nationale Supérieure d'Arts et Métiers, 2002.

[34] J. Moré and D. Sorensen. Computing a trust region step. *SIAM J. Sci. and Stat. Comput.*, 4(3):553–572, 1983.

[35] Hasan Ocak and Kenneth A. Loparo. Estimation of the running speed and bearing defect frequencies of an induction motor from vibration data. *Mechanical Systems and Signal Processing*, 18:515–533, May 2004.

[36] J.G. Proakis and D.G. Manolakis. *Digital Signal Processing: Principles, Algorithms, and Applications*. Englewood Cliffs, NJ: Prentice Hall, 1996.

[37] J. Kørte. An approach to management of health monitoring applications. *Journal of Quality in Maintenance and Engineering*, 2002.

[38] J. Kørte and T. Aven. Health monitoring of helicopters - a framework for decision making based on health information. *Foresight and Precaution (Proceedings of Esrel 2000)*, 2000.

[39] B. Samanta and K. R. Al-Balushi. Artificial neural network based fault diagnostics of rolling elements bearings using time-domain features. *Mechanical Systems and Signal Processing*, 17:317–328, March 2003.

[40] P.D. Samuel and D.J. Pines. Planetary gear box diagnostics using adaptive vibration signal representations: A proposed methodology. In *American Helicopter Society 57th Annual Forum*, Washington, DC, USA, 2001.

[41] P.D. Samuel and D.J. Pines. Helicopter transmission diagnostics using constrained adaptive lifting. In *American Helicopter Society 59th Annual Forum*, Phoenix, Arizona, USA, 2003.

[42] R.M. Stewart. Some useful analysis techniques for gearbox diagnostics, technical report mhm/r/10/77. Technical report, Machine Health Monitoring Group, Institute of Sound and Vibration Research, University of Southampton, 1977.

[43] Ronald M. Stewart. Advanced transmission system health monitoring (surface space technology). In *European Helicopter Association Conference*, 1987.

[44] N.S. Swansson. Application of vibration signal analysis techniques to health monitoring. In *Conference on Friction and Wear in Engineering*, 1980.

[45] Wim Sweldens. The lifting scheme: A custom-design construction of biorthogonal wavelets. *Applied and Computational Harmonic Analysis*, 3(2):186Ű–200, 1996.

[46] Bang Tran. A rapid helicopter drive train fault detection using adaptive-network-based fuzzy method. In *32nd European Rotorcraft Forum*, Maastricht, Nederland, September 2006.

[47] G. Wackers and J. Kørte. Drift an vulnerability in a complex technical system: Reliability of condition monitoring systems in north sea offshore helicopter transport. *Internal Journal of Engineering Education*, 2003.

[48] Johan Wiig. Methods for parametric trend analysis. In *JDL6'06*, Giens, France, June 2006.

[49] Johan Wiig. Structuring and communication of hums data. In *32nd European Rotorcraft Forum*, Maastricht, Nederland, September 2006.

[50] Johan Wiig and Hassan Noura. Flight stage normalization of hums indicators. In *American Helicopter Society's 62nd Annual Forum*, Phoenix, Arizona, USA, May 2006.

[51] Johan Wiig, Hassan Noura, Daniel Brun-Picard, and Jean-Charles Anifrani. Trend analysis of helicopter condition monitoring indicators. In *American Control Conference*, Minneapolis, Minnesota, USA, June 2006.

[52] Johan Wiig, Hassan Noura, Daniel Brun-Picard, and Jean-Pierre Derain. Trend-based transmission system diagnosis. In *45th IEEE Conference on Decision and Control*, San Diego, California, USA, December 2006.

[53] J.J. Zakrajsek. A review of transmission diagnostics research at nasa lewis research center, technical report nasa tm-106746, arl-tr-599. Technical report, NASA and the US Army Research Laboratory, 1994.

[54] J.J. Zakrajsek. Transmission diagnostics research at nasa lewis research center, technical report nasa tm-106901, arl-tr-748. Technical report, NASA and the US Army Research Laboratory, 1994.

[55] J.J. Zakrajsek, D.P. Townsend, and H.J. Decker. An analysis of gear fault detection methods as applied to pitting fatigue failure data, technical report nasa tm-105950, avscom tr-92-c-035. Technical report, NASA and the US Army Aviation Systems Command, 1993.

Résumé

Le système de surveillance (HUMS) installé dans les hélicoptères permet d'anticiper les anomalies et de donner la possibilité d'effectuer des tâches de maintenance prédictive avant l'apparition de défauts critiques. Par ailleurs, HUMS est également destiné à détecter la propagation de défauts émergents. Ceci consiste à comparer les caractéristiques vibratoires en vol de l'hélicoptère aux caractéristiques d'un état normal prédéfini. L'inconvénient majeur de cette approche est que les caractéristiques de l'état normal sont relatives au type de l'hélicoptère et changent après les tâches de révision et de maintenance, ce qui nécessite un réapprentissage de ces caractéristiques.

Cette étude présente des méthodes d'évaluation de la progression temporelle des signatures vibratoires. L'étude de l'évolution de la signature vibratoire dans le temps permet de détecter des événements comme des interventions de maintenance ou des propagations de défauts sans avoir à définir un modèle de l'état de bon fonctionnement de l'appareil. Des méthodes fondées sur des modèles paramétriques et des bancs de filtres d'analyse vibratoire ont été testées et validées. Finalement, une méthode de détection de défauts a été mise en œuvre et a donné de meilleurs résultats que les méthodes traditionnelles utilisées.

Mots-clés : HUMS, Diagnostic de défauts, Analyse vibratoire, Analyse de tendance

---

Abstract

A Health and usage Monitoring System (HUMS) anticipates discrepancies in the rotorcraft drive-train, giving the operator an opportunity to perform corrective maintenance before any damage becomes critical. In addition to usage spectrum analysis, a HUMS deploys vibration monitoring as a means to detect propagating faults. This method consists in comparing in-flight vibration recordings to a normal state baseline. A recurrent problem with this approach is that this baseline is aircraft specific and subject to change between major overhauls, forcing the operator to relearn the baseline on regular bases.

This study presents methods for evaluating the time-progression of the drive-train vibration signature. By studying fluctuation in vibration signature over time, it is possible to detect events such as maintenance actions and fault propagations without any aircraft specific baseline. Several progression analysis methods are tested, both parametric models and filter-banks. Finally, progression analysis is used as a basis for fault detection, and is shown to produce better results than traditional methods.

Keywords: HUMS, Fault diagnosis, Vibration monitoring, Trend analysis