Development of a Machine Learning Tool as a Predictive Maintenance Solution for the Flight Control System of the Embraer E190

Nirina Sofia Leão Raharitahiana

Thesis to obtain the Master of Science Degree in

Aerospace Engineering

Supervisors: Prof. Sérgio David Parreirinha Carvalho
Eng. Pedro Alves Toscano Antunes Figueira

Examination Committee

Chairperson: Prof. Fernando José Parracho Lau
Supervisor: Prof. Sérgio David Parreirinha Carvalho
Member of the Committee: Prof. José Raul Carreira Azinheira

January 2021

To my family, who will always be there for me.

Acknowledgements

Nothing worth having in life comes easily, and the process of learning is certainly a path paved in cold, hard stone. Were it not for the most special people in my life, I would not have come this far in that track.

I would like to first share my deepest thanks to my supervisors, Professor Sérgio Carvalho and Engineer Pedro Figueira, for allowing me the opportunity to work alongside Portugália Airlines to deliver this project. A kind word of appreciation for Professor Sérgio Carvalho for his never-ending understanding, patience and humanity.

Moreover, this acknowledgement would never be complete were I not to express my tremendous gratitude towards Engineer Nuno Lima from Portugália Airlines, for his patience in guiding me, insightful advice whenever inquired, and generous availability to discuss the issues at hand. This thanks extends to everyone I met at Portugália Airlines, for always making me feel welcome and in a warm environment, despite having the opportunities to be in the office been so scarce due to the COVID-19 pandemic.

To my brother in arms, best friend throughout this path; many miles may separate us now, but we always have each other's back.

Finally, I wish to thank my family, for always providing the necessary support in every possible way. Without them, none of this would have been achieved.

Resumo

Numa realidade em que a procura de transportes aéreos caiu abruptamente derivando de uma situação sem precedentes, num mercado que já se adaptara à alta procura, e era já extremamente competitivo, é necessário reduzir os custos operacionais para sobreviver num sector mutilado por estas circunstâncias exteriores. Uma das áreas em que esta redução tem maior potencial corresponde à área da manutenção. Avanços nas tecnologias que facilitam a obtenção de dados de voo, e novas soluções emergentes de aprendizagem automática possibilitam agora a introdução de novos métodos para abordar os problemas derivados de falhas em componentes da aeronave, que causam não só atrasos, como muitas vezes a impossibilidade de proceder o voo, causando a condição de *Aircraft-on-Ground* (AOG), o que leva invariavelmente a consequências financeiras altas.

Este projecto teve por objectivo a criação de uma solução de aprendizagem automática para a manutenção preditiva correspondente ao Sistema de Controlos de Voo da frota da Portugália Airlines, composta por 13 aeronaves da Embraer. Para tal, foram usados os dados vindos dos vários sensores das ditas aeronaves, mensagens de alerta gerados pelas mesmas, e os relatórios provenientes das equipas de manutenção.

Várias medidas foram criadas para relatar o nível de degradação do sistema, de modo a fornecer ao modelo criado a informação necessária para que pudesse estimar, com base em voos anteriores, quantos voos restam até se dar uma falha no sistema.

Os resultados obtidos mostram o potencial da solução e que o modelo consegue identificar padrões de degradação no sistema.

Palavras-chave: Dados de voo, Manutenção predictiva, Aprendizagem automática, Sistema de controlo de voo

Abstract

In a reality where the demand for air transport has abruptly dropped due to a situation without precedent, in a market which had already adapted to high demand, and which was already extremely competitive, it is necessary to reduce operational costs to survive in a sector which has been mutilated by these exterior circumstances. One of the areas in which this reduction has most potential is maintenance. Technological advancements which facilitate the acquiring of flight data and new emerging machine learning solutions enable the introduction of new methods to address problems deriving from faults in aircraft components, which not only lead to delays, but many times the impossibility of flying, causing a condition known as Aircraft-on-Ground (AOG), inevitably leading to major financial consequences.

This project had as objective the creation of a machine learning solution for the predictive maintenance of the Flight Control System of Portugália Airlines' fleet, comprised of 13 Embraer aeroplanes. For this, flight data from the various aircraft sensors were used, as well as alert messages generated by the aircraft's systems, and the reports provided by the maintenance teams.

Several variables were created to model the degradation level of the system, so as to provide the necessary information to the created model, so it could estimate how many flights the system has until it is likely to experience a fault, based on previous flights.

The results show the potential of the solution, and that the model succeeds in identifying degradation patterns in the system.

Keywords: Flight data, Predictive maintenance, Machine learning, Flight control system

Contents

List of Tables

List of Figures

Nomenclature

Greek symbols

α Parameter controlling the minimum similarity between curves that an estimate is allowed to have in order to be used in the final estimation calculation.

λ Parameter controlling the scale of the similarity between two curves.

Roman symbols

$H^{(u)}$ Degradation Curve for sample sequence u.

$e_t^{(u)}$ Reconstruction error at time t for sequence u.

h_t Health index at time t.

$\hat{R}^{(u)}$ Estimation of the remaining useful life of sample sequence u.

$R^{(u)}$ Remaining useful life of sample sequence u.

s Similarity between curves.

t Time/time-lag.

t_{max} Maximum time lag.

u Sample sequence.

Abbreviations

AGS Analysis Ground Station.

AOG Aircraft-on-Ground.

ATA American Air Transport Association.

FS-ACE Flap/Slat Actuator Control Electronics.

HS-ACE Horizontal Stabiliser Actuator Control Electronics.

P-ACE Primary Actuator Control Electronics.

S-ACE Spoiler Actuator Control Electronics.

CAN	Controller Area Network.
CAS	Crew Alerting System.
CMC	Central Maintenance Computer.
FCM	Flight Control Module.
FHDB	Fault History Database.
FPDU	Flap Power Drive Unit.
GRU	Gaited Recurrent Unit.
HI	Health Index.
HSTA	Horizontal Stabiliser Trim Actuator.
LRM	Line Replaceable Module.
LSTM	Long-Short Term Memory.
MAE	Mean Absolute Error.
MAU	Modular Avionics Unit.
MRO	Maintenance, Repair and Overhaul.
NN	Neural Network.
RNN	Recurrent Neural Network.
PCU	Power Control Unit.
PDPU	Power Down/ Power Up.
PM	Predictive Maintenance.
RMSE	Root Mean Squared Error.
RTS	Return to Service.
RUL	Remaining Useful Life.

Chapter 1

Introduction

The following sections discuss the motivation behind this project, with a brief introduction to the maintenance practises in the aviation industry, along with rising tools to make said maintenance more efficient. After that, the objectives of the project are stated, and the structure of the text of this work is given.

1.1 Motivation

In an age of generally ever growing demand for air transportation (see Fig. 1.1), yet along with a general decrease in cost of airfares, and especially now with the crisis of the COVID-19 pandemic which took a blow at the aeronautical sector worldwide, it becomes increasingly important to decrease operational costs.

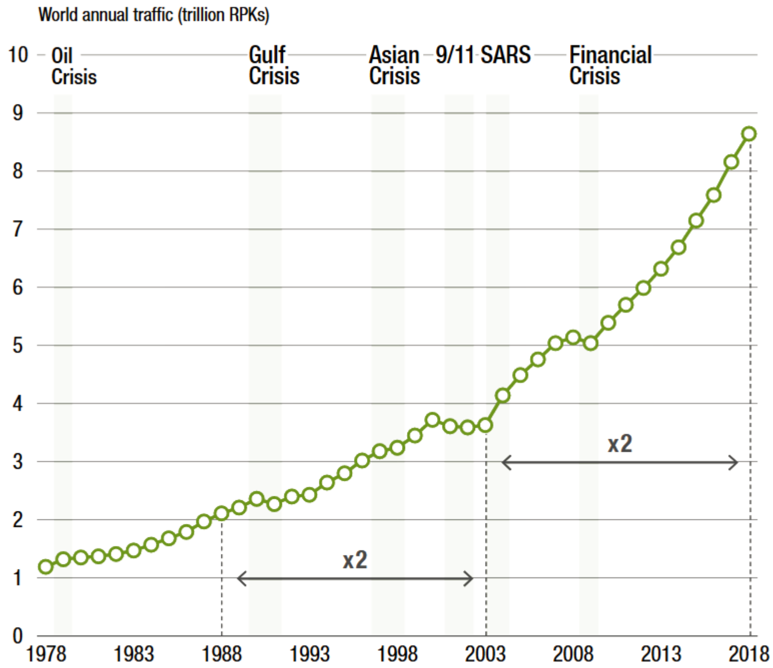


Figure 1.1: World annual traffic from 1978 to 2018, in Revenue Passenger Kilometers (RPK) [1]

Maintenance is a major factor in that regard, with the global Maintenance, Repair and Overhaul (MRO) spend in 2018 being valued at $69 billion, representing 9% of airlines operational costs [2]. In fact, maintenance improvements have been stated as being one of the top three savings for airlines, these savings listed as [2]:

- Health monitoring and predictive maintenance driven by improved dispatch reliability, labour productivity,
- Fuel cost savings, and
- Delay reduction through improved turnaround process.

According to [3], the industry definition of maintenance costs is the spend derived from restoring or maintaining the systems, components, and structures of an aeroplane in an air-worthy condition; it can be divided in two kinds of cost: direct airframe and engine maintenance, and maintenance overhead. The former kind pertains to all costs concerning the "servicing, repair, modification, restoration, inspection, test, and troubleshooting tasks during on-aeroplane and shop maintenance activities"; while the latter kind pertains to "unallocated labour costs and expenses for maintenance supervision, training, and planning, equipment rental and utilities".

The maintenance actions of interest to this work may be classified as one of the four main types: corrective, preventive, condition monitoring, and predictive maintenance.

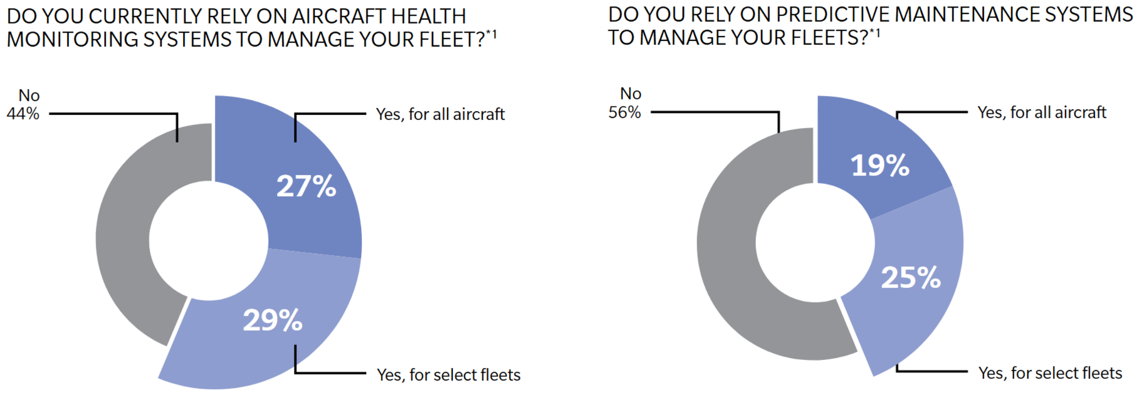
Corrective maintenance, also possibly referred to as run-to-failure maintenance, is the act of repairing a certain component only after a failure on it has occurred. While it may be the ideal system for "low-priority equipment, without which the company's operations can continue running normally" [4], it becomes undesirable when it pertains to critical or otherwise generally necessary components.

Preventive maintenance, on the other hand, involves a scheduled, regularly performed activity on a component with the objective of bettering its chances of not failing. The scheduling of this maintenance type may be done under either of two principles: time-based (i.e. at a fixed period) or usage-based (e.g. at a fixed mileage on a vehicle). Preventive maintenance is recommended when a component has an increasing probability of failure over time; if its failures are random, preventive maintenance is futile.

Condition monitoring involves the assessing of a component's reliability based on available operation data, attempting to identify wear or degradation in order to act accordingly.

Finally, predictive maintenance is a step up from condition monitoring and regards acting to prevent a failure based on knowledge of when it is going to occur. In other words, data pertaining to the component at hand may be used to apply machine learning and analytics to assess its reliability, thus performing maintenance actions accordingly and as needed. As more and more data becomes available, there is a gradual trend in adopting this technique to avoid the shortcomings of the others.

The aviation industry is no exception to this trend. In fact, many condition monitoring solutions exist (specifically referred to as Aircraft Health Monitoring systems), and more recently an emerging market for predictive maintenance solutions grows with the increasing desire to reduce maintenance costs and improve the provided services. Fig. 1.2 presents the results of a survey to study the adoption of these maintenance solutions by the surveyed airlines, showing that over half of these airlines had already Health Monitoring Systems in use, and, concerning Preventive Maintenance, while it had not been as widely adopted as the Health Monitoring Systems, already a very high portion of the airlines relied on it.



(a)

(b)

Figure 1.2: Adoption of Aircraft Health monitoring and Preventive Maintenance Systems, according to a survey by Oliver Wyman [5]

Aircraft manufacturers already offer health monitoring services. AIRcraft Maintenance ANalisys (AIRMAN) is Airbus' solution [6], used by 140 costumers on around 7 000 aircraft as of 2016. It provides a constant monitoring of the aircraft's system's health and transmits important fault and warning messages to ground control. Boeing also possesses a service in this area, the Airplane Health Management (AHM) service [7], which also collects data from the fleet and sends it to ground control.

The manufacturer of the aircraft in analysis, Embraer, also provides a similar service: the AHEAD-PRO, which provides aircraft usage information, while sending emerging error messages of the aircraft's systems in real time to the operation centre, providing with them the likely causes of such messages.

Data generated by the aircraft in Portugália Airlines' fleet can reach several megabytes per flight. This data has the potential of being used not only in a health monitoring tool, but also a predictive maintenance one, which is where the machine learning solution of this work finds its use.

Machine learning has become an ever so present concept in the world we live in and so many technological advancements we rely on today. Many times, we do not realise what is under the hood, what engine is driving some piece of technology; but most times we do not even so much as realise, or care to notice, we are using it. Simply typing in a query on a web search engine and being presented search suggestions is a product of machine learning advancements. Likewise, having videos, movies or series which we are likely to enjoy being automatically recommended to us is a product of machine learning advancements. Text-to-speech, translation software, text recognition and face recognition in images are all possible thanks to machine learning.

And this is so as machine learning is such a versatile concept, expandable to a plethora of subjects, because instead of relying on programmers hard-coding ways in which an algorithm should behave, the machine is expected to learn by itself how to deal with a certain input. This represents an obvious advantage to traditional programming: it would be impossible to input every possible face into an algorithm so as to have it recognise faces in photographs, whereas with machine learning, the program is required to learn the general features of a face, and apply that knowledge into some other example, being then able to tell, with some degree of certainty, whether or not there is a face in the picture.

With such potential, it would only be a waste to not try to apply machine learning to engineering problems. And indeed, “machine learning is becoming a driving force in the field of industry-grade predictions, delivering significantly more reliable forecasts than traditional statistical methods, particularly where there is access to vast quantities of ‘unstructured’ data” [8].

1.2 Objectives

The objective of this dissertation is to provide Portugália Airlines with a machine learning solution to aid in applying predictive maintenance on the Flight Control System of the aircraft of their fleet. The solution should be able to output an estimated time until the next failure on this system, taking advantage of the flight data generated by each aircraft, so that maintenance may rely less on preventive and especially corrective methods, hence becoming more efficient and less costly.

1.3 Thesis Outline

This work is divided in 6 chapters: Introduction, Fundamentals, Methodology, Results, Result Discussion, and Conclusion.

The Introduction chapter aimed to establish the objectives of the project, along with their motivation, as well as establishing a structure for this text.

The Fundamentals chapter is meant to introduce the main theory of this text to the reader, so as to provide an easier understanding of the terminology, and the workings of the concepts which are the basis of the project. Here, an brief overview of the studied aircraft model and its Flight Control System is made, along with the data that were used to arrive at the desired model; an introduction to some relevant machine learning fundamentals is also made.

The Methodology chapter discusses and presents the methods used to arrive at the wanted solution, going through data gathering, data pre-processing, machine learning model used, and data post-processing to achieve the results.

After this, said results are presented in the Results chapter, where two different sets of Remaining Useful Life predictions are shown, the second set being an attempt at improving the first set.

Following, the Result Discussion chapter analyses the obtained predictions for each set, discussing validity and generalisability, and finally exploring the obstacles found on the implementation, along with what could be done to improve it.

Finally, the Conclusion chapter closes the text with the discussion of the achievements of this work, along with possible future work to be done as an extension of this.

Chapter 2

Fundamentals

With the set objective of studying failure predictability on the Flight Control System of the Portugália Airlines' fleet via a machine learning solution, some basic fundamentals should be introduced in order to have a full grasp on the intricacies of the task.

Following is an introduction to the studied aircraft, along with an overview of the Flight Control System which should acquaint the reader with the basic workings of each subsystem, along with their main components, and some terminology. After this, the available data are presented, and a brief discussion on the relevant machine learning concepts is had in order to introduce the terms later used in the Methodology chapter.

2.1 Embraer E190

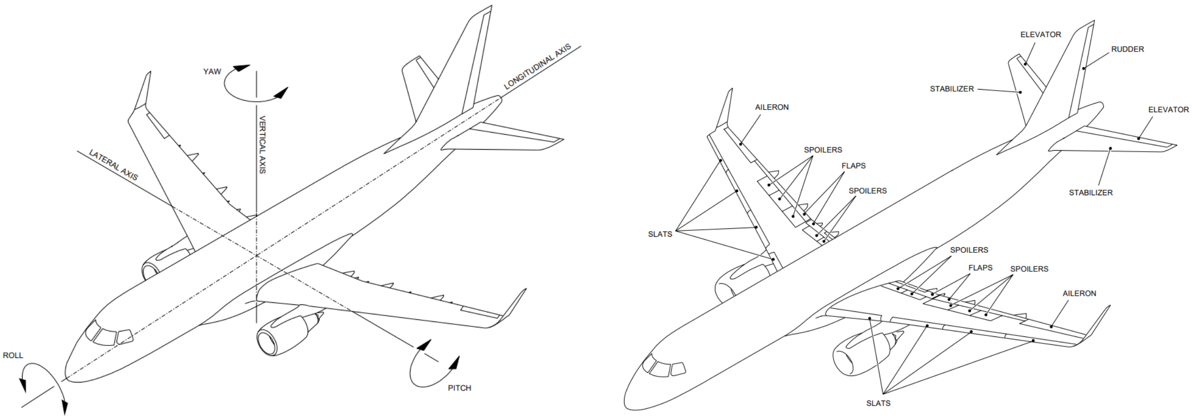
The Embraer E190 is part of the Embraer E-Jet family, which is comprised of four different aircraft models — E170, E175, E190, and E195 — all of which narrow-body twin engine aircraft capable of short to medium range flights, and with a seating capacity of 70 to 130 seats, being marketed by Embraer as regional aircraft with 'the big jet feel'. [9]

Portugália Airlines' fleet is currently composed of nine Embraer E190 and 4 Embraer E195 [10]. These two variants differ from each other in fuselage length (thus seating capacity) and not much else. In fact, in terms of the Flight Control System, and this work, both models are the same, hence treated the same. Table 2.1 shows the general specifications of the E190 model.

Table 2.1: Embraer E190 general specifications [11]

Engine	2 × General Electric CF34-10E
Power	2 × 20 000 lbf
Avionics	Honeywell Primus Epic EFIS
Max Cruise Speed	Mach 0.82
Service Ceilling	41 000 ft
Range	4 537 km
Seating Capacity	106 on a 2-class layout
Fuselage Length	36.24 m
Fuselage Diameter	3.01 m
Wingspan	28.27 m
Maximum Take Off Weight	50 300 kg

2.2 Flight Control System – An Overview



(a) Lateral, longitudinal, and vertical axes on the aircraft (b) Flight control system surfaces on the aircraft

Figure 2.1: Flight Control System overview [12]

According to Airlines for America, formerly known as the Air Transport Association of America (ATA), the Flight Control System belongs to the ATA chapter 27 [13] (henceforth ATA 27, for short), which is a numbering system standard for all commercial aircraft documentation [14].

The flight control system is the system responsible for allowing directional control of the aircraft, being comprised of surfaces on wings and tail, and being made up of primary and secondary flight control systems [12]. The primary controls allow control of the aircraft about the lateral, longitudinal, and vertical axes (see Fig. 2.1a) and include the following subsystems: Ailerons, Elevators, Multifunction Spoilers, and Rudder.

The secondary controls serve as aid in lift generation and handling of the aircraft. These include the Flaps, Slats, Spoilers/Speed Brakes, and Horizontal Stabiliser subsystems. All these surfaces are represented in Fig. 2.1b.

Additionally, an Electrical System is responsible for operating the electronically controlled fly-by-wire system.

All control surfaces have surface position sensors which provide information on their deflection. Additionally, the flaps and slats also have skew sensors which are used to prevent a skew condition from occurring. A skew condition pertains to when either the inboard or outboard edge of the flap moves further than the other [15]; should this happen, the skew sensors detect this, and the flap is disallowed from moving further.

In the next sections, a brief description of each of the aforementioned subsystems is given.

2.2.1 Electrical System

The Flight Controls Electrical System is responsible for operating the fly-by-wire technology used to allow pilot control of most of the control surfaces, and it should not be mistaken for the aircraft's main Electrical System, which is not a subject of this project, as it pertains to the ATA chapter 24 [13]. Further mentions of "Electrical System" in this work henceforth imply "Flight Controls Electrical System". Fig. 2.2 illustrates this system's components and how they interface, which is discussed in the following paragraphs.

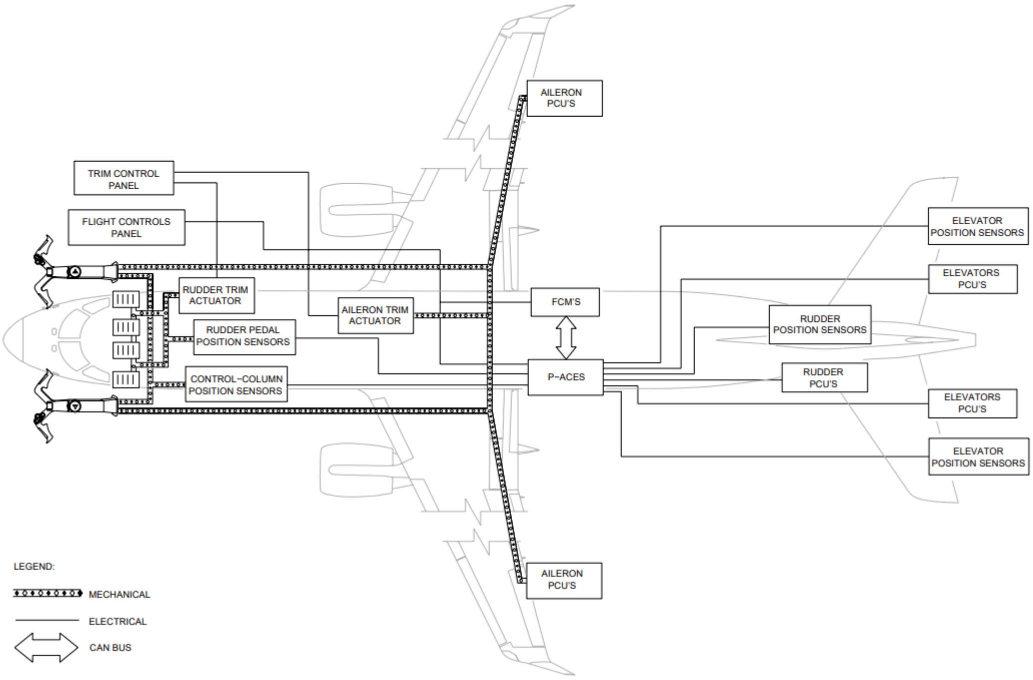


Figure 2.2: Flight Control Electrical System components interface [12]

The Flight Controls Electrical System is comprised of a flight control panel, four Flight Control Modules (FCMs), three Primary Actuator Control Electronics (P-ACEs), a trim control panel, and a fly-by-wire backup battery [12]. Of these, the FCMs and P-ACEs will be the most mentioned in this text, as they are the source of most Electrical System faults.

The P-ACEs are analog controllers providing a direct electronic path between the cockpit control position sensors and the actuators which power the primary control surfaces. Each P-ACE includes two

ACE channels, A and B. [12]

There are a total of four FCMs, each numbered 1 through 4 accordingly. An FCM is a dual slot wide module housed in the Modular Avionics Unit (MAU). A MAU is a cabinet system holding multiple Line Replaceable Modules (LRMs), supplying instalation space, power sources, and the necesseray aircraft interfaces required for those LRMs. There are three MAUs in an aircraft, only two of them housing FCMs, MAU1 and MAU3: MAU1 houses FCMs 1 and 2, and MAU3 houses FCMs 3 and 4. The FCMs are primarly digital controllers which provide the digital interface between the Flight Control System and the aircraft's avionics systems. Furthermore, they provide digital control inputs to the P-ACEs, by means of a Controller Area Network (CAN) bus interface, to augment the cockpit control inputs. [12]

2.2.2 Ailerons

The Ailerons are the only primary control surfaces not operated via fly-by-wire technology. Instead, movement on the control yoke is transmitted to the surfaces via a hydromechanical control system: movement on either of the two mechanically-connected control yokes (pilot or copilot) is transmitted to the torque tubes, then to the cable circuit comprised of corrosion-resistant steel cables, pulleys and quadrants, and eventually this motion is transmitted by input rods to the hydraulic Power Control Units (PCUs), which in turn move the aileron surfaces. [12]

The most relevant components to the reliability of this subsystem will be shown to be the cables, and the PCUs, the latter not exclusively to the Aileron subsystem.

2.2.3 Elevators and Rudder

Both the elevators and the rudder are controlled via fly-by-wire technology. The positions of their respective controls — control column and pedals — are electronically sensed and processed by the P-ACEs to move the surfaces with the PCUs. Additionally, the FCMs amplify the cockpit controls signal and transmit that to the P-ACEs as well, as discussed before. [12]

2.2.4 Horizontal Stabiliser

The horizontal stabiliser is moved by a Horizontal Stabiliser Trim Actuator's (HSTA's) one of two brushless motors. The Horizontal Stabiliser Actuator Control Electronics (HS-ACE) receives trim signals from either an FCM via a CAN bus interface, or the cockpit's trim switches, and in turn sends an input to the HSTA. [12]

2.2.5 Flaps and Slats

The flaps are electronically commanded via the Slat/Flap Control Lever. When the lever is moved from one position to another, the Flap/Slat Actuator Control Electronics (FS-ACE) commands the Flap Power Drive Unit (FPDU), which outputs a rotary motion. This motion is transmitted along the wing by a mechanical drive line to the flap actuators, which convert it into a linear motion to move the flap panels [12]. The slats work analogously to the flaps.

As will later be discussed, the most prevalent failure modes on the Flap and Slat subsystems pertain to failure of the position sensors, skew sensors, and actuators.

2.2.6 Spoilers and Speed Brakes

The Spoiler and Speed Brake subsystem is comprised of both Ground Spoilers and Multifunction Spoilers.

Ground Spoilers, as the name suggests, are only used on the ground, which is to say, when the condition “weight-on-wheels” is fulfilled, and theirs is a single function: to decrease the required stopping distance of the aircraft after touchdown, during the roll-out. These spoilers are deployed automatically after touchdown by the four FCMs, and there is no other means of input. [12]

In the available dataset, the Ground Spoilers were not found to be the source of any fault, neither are there much relevant sensor data on them, so they are henceforth neglected toward the system analysis. For this reason, any further mention of simply “Spoilers” should be understood as “Multifunction Spoilers”.

The Multifunction Spoilers have four separate functions [12]:

- Roll augmentation to aid the function of the Ailerons, where they are proportionally commanded by the control wheel’s displacement, as an asymmetrical deployment;
- Lift reduction during flight, controlled by the Speed Brake Handle, and deployed symmetrically;
- During roll-out after touchdown deployed symmetrically with the Ground Spoilers, and
- During steep approach mode, deployed symmetrically.

Each of the six multifunction spoiler panels is controlled by a PCU. Because the Spoiler Actuator Control Electronics (S-ACE) reside within the FCMs, these PCUs receive input from FCMs 1, 3 and 4, which in turn receive input from the roll commands and the speed brake handle. [12]

2.3 Available Data

The available data used in this work pertains to three different sources: flight data consisting of sensor data from the aircraft, maintenance and crew alerting messages generated by the aircraft’s Central Maintenance Computer (CMC), and maintenance reports.

The sensor data was obtained via the company Sagem’s Analysis Ground Station (AGS) software, which is the flight analysis software tool used by Portugália Airlines. After exporting, the data consisted of files — a file per flight — with sensor readings of the Flight Control System throughout the flight.

The event of failure was defined as the appearance of the message FLT CTRL NO DISPATCH in the Crew-Alerting System (CAS), which is a cautionary message which dictates immediate maintenance intervention, without which the aircraft is not allowed to fly.

Messages generated by the aircraft's CMC were accessed via the Fault History Database (FHDB). These pertain to maintenance messages generated by the aircraft when it is in operation, indicating alerts on its various systems.

Finally, the maintenance reports were accessible via the AMOS software, the MRO solution in use by Portugália Airlines.

2.4 Relevant Machine Learning Concepts

In this section, the relevant machine learning concepts to this work are introduced.

2.4.1 Supervised and Unsupervised Learning

There are two main categories in machine learning pertaining to the nature of the input, and which influence how an algorithm may or may not be developed. These are supervised and unsupervised learning. [16]

Supervised learning pertains to when the data from which the algorithm will learn (train data) contains both the defining variables of the data (the features) and the known result that that combination of variables yields (the labels). That is to say that, for instance, there could be a dataset consisting of cars, whose features might be horsepower, mileage, consumption, number of doors, etc., while the labels might be the prices of said cars. So, for each combination of features given to the algorithm, the resulting price is also given. The algorithm would then learn from the dataset and, given a test example it has never seen before, it should be able to output an estimated price for that car based on its previous given knowledge.

On the other hand, when the data consists of only features and no labels, the corresponding algorithm would be an unsupervised learning algorithm.

2.4.2 Regression and Classification Problems

Depending on the possible output extracted from the machine learning model, a problem may be classified as either a Regression problem, or a Classification problem [17]. Going back to the car dataset example, the output of the algorithm is the price of a car, which could be any value within a continuous range. This is therefore a Regression problem.

On the other hand, given for instance a dataset of flowers, whose features might be the physical characteristics of each sample flower, such as height, petal colour, petal length, etc., an algorithm might be tasked with determining the species of flower that is presented to it. In this case, there are only so many values (categories) which the output can present, that is to say that there is a discrete range of values for the output. This is then referred to as a Classification problem.

2.4.3 Time Series Processing Models

The dataset pertaining to the problem at hand consists of sensor data which characterises the functioning of the control system of the aircraft throughout time, and this therefore consists of time series data. Not all machine learning solutions are suited for this type of data. For example, a regular Neural Network (NN) takes as input a fixed size vector which is limiting when dealing with a series data input is needed. One workaround would be to take each step of the series and call the NN repeatedly. However, this approach would completely disregard the fact that every input of the series depends on what precedes it: there is no sense of time dependency.

The solution to this issue lies in a variant of the NN called Recurrent Neural Network (RNN). RNNs have the ability to remember the past and are therefore capable of capturing the relationships between the current input in the series and what happened in the past. However, despite being able to remember the past, an RNN suffers from the issue of short-term memory. Given a long input series, the RNN might be able to relate the current input with its neighbours but cannot relate it to elements which are further away in the series. This leads to an RNN potentially forgetting useful information simply due to the series being too long. The solution to this is using instead models which have longer memory, such as Long Short-Term Memory (LSTM) or Gated Recurrent Units (GRU).

2.4.4 LSTM Layer

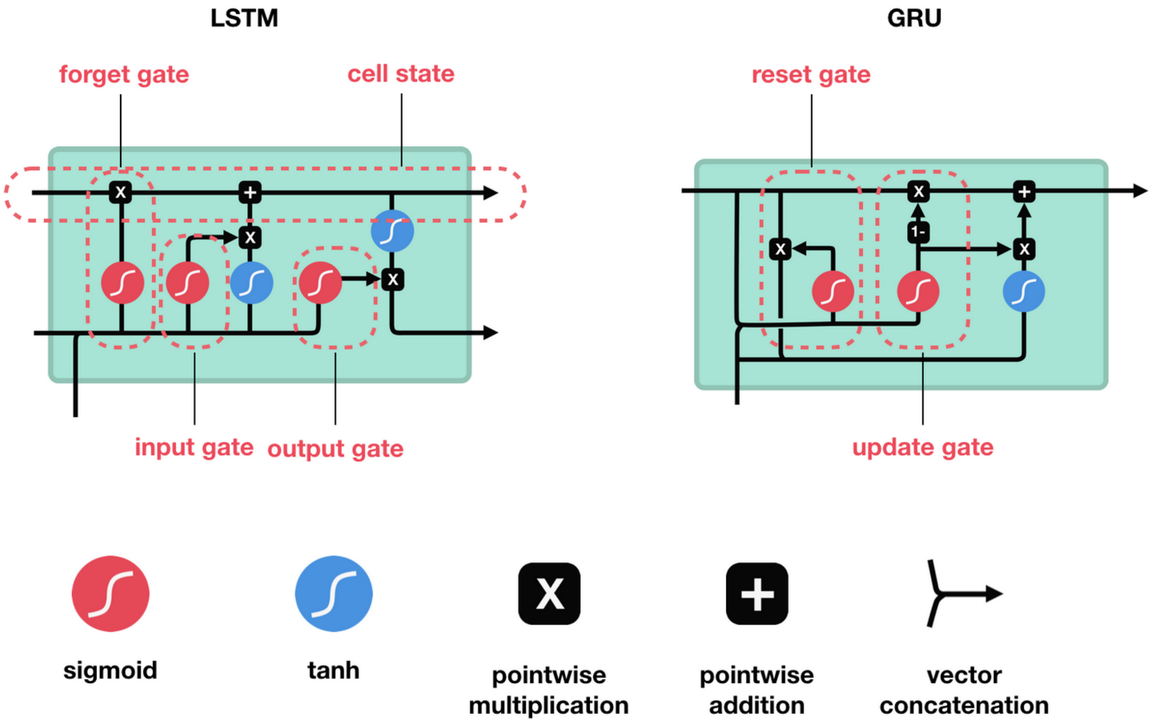


Figure 2.3: Architecture of an LSTM unit (left) and a GRU (right) [18]

The way the data is processed in a regular RNN is that each cell receives an input from the data and the state vector of the preceding cell. These two vectors are concatenated and transformed with a tanh function. With this, the cell creates its own state vector and passes it down the next cell. As stated before, this implementation suffers from short-term memory.

The way this problem is circumvented is by introducing gates in the cell (see Fig. 2.3). In an LSTM unit, each cell has gates, whose job is to learn which of the data from the preceding cell are important to keep, and which can be safely thrown away.

2.5 The Autoencoder

Of specific interest to this work is the concept of Autoencoder. It revolves around an algorithm learning to encode a signal into a latent space, in practicality compressing the signal, and its applicability is surprisingly large, especially with respect to anomaly detection (as examples, see [19, 20]).

2.5.1 Functioning of an Autoencoder

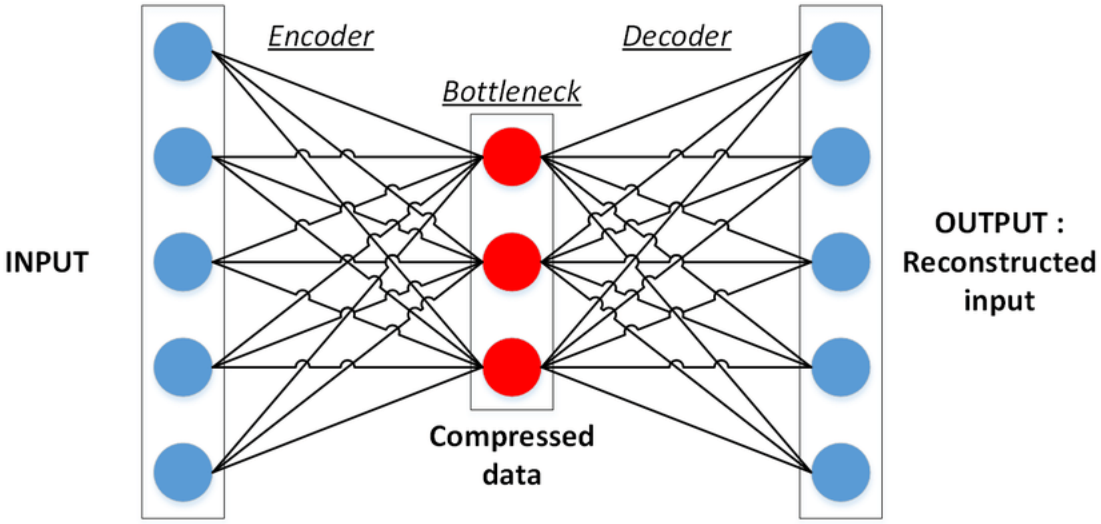


Figure 2.4: Basic architecture of an autoencoder [21]

To have an Autoencoder learn to encode a signal, it is necessary to have both an Encoder and a Decoder (see Fig. 2.4). A signal is input in the Encoder, which compresses it and feeds it to the Decoder, which in turn tries to reconstruct the signal back to its original state. In the end of this cycle, the algorithm measures how close it was to the original signal and the reconstruction error is used to iterate again, repeating the cycle until the desired accuracy is achieved. At this point, the Encoder can look at the most relevant characteristics of the input signal and construct a good encoding of said signal.

2.5.2 Autoencoders in Anomaly Detection

The most common way of making use of the concept of Autoencoder in anomaly detection is via its reconstruction error. The main idea behind this is that if the Autoencoder is trained to be able to encode and decode back (read reconstruct) normal data, then, when tasked with reconstructing anomalous data, the reconstruction error will be greater. This error can therefore be used as a measure of how anomalous the data is.

2.6 Health Indicator

Many methods to estimate the Remaining Useful Life of a component revolve around the calculation of a Health Index (HI). As the nomenclature suggests, this is a numerical value representative of the system's health. There are three types of health index, depending on how or on what basis it is calculated [22]:

- The *Physical Health Index* is defined from physical characteristics or parameters of the current state of the component and its operation, such as cracks;
- the *Probabilistic Health Index* is defined by the probability of the component being in a healthy state, or its reliability; its value ranges from 0 to 1, 1 being the best possible state;
- the *Mathematical Health Index* has only mathematical meaning and it can present any value. This is one of the most prevalent Health Index types in Remaining Useful Life estimation via machine learning methods, where HI is defined by the algorithm from a set of data.

Chapter 3

Methodology

The purpose of this work is to obtain an estimate of how long a certain aircraft has until it is likely to experience an event of failure of the Flight Control System, based on the sensor data available. The machine learning solution chosen as the basis for this was proposed in [19], where an LSTM-Autoencoder was employed to try to determine the remaining useful lives of several instances in two different publicly available datasets: the C-MAPSS Turbofan Engine Dataset [23] and the Milling Machine Dataset [23].

Furthermore, the methodology used in the present work to apply this algorithm to the specific problem can be divided into four different stages:

- Data gathering — feature creation;
- Data pre-processing and feature selection;
- Autoencoder, and
- Data post-processing.

In the subsequent subsections, an overview of the used algorithm will be made, followed by a description of each of these four stages.

3.1 Algorithm Overview

Given the system of interest's run-to-failure sequences of sensor data over time, an Autoencoder is made use of to try to predict the Remaining Useful Life of a given test instance which has not yet failed. A run-to-failure sequence should be understood, in this context, as the evolution, along time, of a set of variables chosen to represent the system's health: each sequence begins at a healthy state and ends at the instant where there is a failure.

For this purpose, first, the Autoencoder is trained to encode and decode again (i.e. reconstruct) healthy data only, which is to say, as an approximation, data where it is known a failure will not occur in a while. To achieve this, it is assumed that the initial cycles of each sequence pertain to healthy data, so the Autoencoder is trained with those only.

Having a trained Autoencoder, it is then tasked with reconstructing the full sequences of the train data (which is to say, the full run-to-failure sequences, with both healthy cycles and degraded cycles). Because it was trained on healthy data only, and these full sequences represent a system that is degrading over time, it is presumable that the Autoencoder will make an ever-increasing error throughout the reconstruction of each sequence. This error is then used as a measurement of the health of the system, and so a normalisation of it is used as a Health Index. So, for each cycle t of the sequence u, the error $e_t^{(u)}$ is normalised as [19]:

$$h_t^{(u)} = \frac{e_M^{(u)} - e_t^{(u)}}{e_M^{(u)} - e_m^{(u)}}, \tag{3.1}$$

where $e_M^{(u)}$ and $e_m^{(u)}$ are the maximum and minimum reconstruction errors obtained for sequence u, respectively.

The result of calculating this Health Index for a sequence u throughout its length $L^{(u)}$ is a Degradation Curve $H^{(u)}$ which represents the degrading health of the system, from healthy state to failure such that $H^{(u)} = [h_1^{(u)}\, h_2^{(u)} ...\, h_t^{(u)} ...\, h_{L^{(u)}}^{(u)}]$. The Degradation Curves obtained from the train data are then stored.

Following, the same curves are obtained for the test data. Because the test data represents instances which have not yet run into a failure, the resulting Degradation Curves are considered incomplete. It is by comparing — matching — the test Degradation Curves with the train Degradation Curves that an estimation of the RUL for a given test instance is made, as represented by Fig. 3.1, showing that that estimation is obtained by varying a time-lag (horizontal offset) between the two curves and calculating the RUL as the remaining time cycles on the train curve after the last cycle of the test curve.

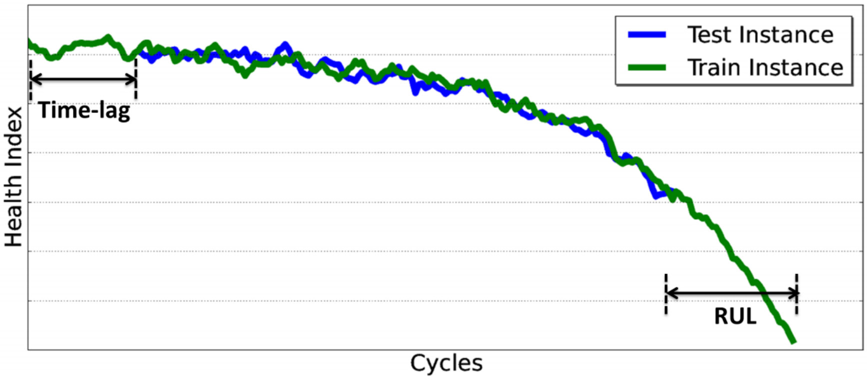


Figure 3.1: Example of RUL estimation using Degradation Curve matching [19]

There being various train curves and various possible values for the time-lag presented in Fig. 3.1, there are multiple estimates for the RUL of a single test instance, so a weighted mean of these estimations is taken as the final predicted RUL. The weights for each estimate are given by the similarity s between each test curve u^* and train curve u for given values of time-lag t, which is computed via the following expression [19]:

$$s(u^*, u, t) = \exp(-d^2(u^*, u, t)/\lambda), \tag{3.2}$$

where

$$d^2(u^*, u, t) = \frac{1}{L^{(u^*)}} \sum_{i=1}^{L^{(u^*)}} (h_i^{(u^*)} - h_{i+t}^{(u)})^2 \tag{3.3}$$

is the squared Euclidean distance between $H^{(u^*)}$ in its cycles 1 through $L^{(u^*)}$, and $H^{(u)}$ in its cycles t through $t + L^{(u^*)}$, and

$$\lambda > 0$$

is a parameter controlling the scale of the similarity, smaller values of λ implying larger difference in s, even when d is small.

Thus, each RUL estimate $\hat{R}^{(u^*)}(u, t)$ given by each train instance and each time lag is used to compute the final RUL estimate $\hat{R}^{(u^*)}_{final}$ by:

$$\hat{R}^{(u^*)}_{final} = \frac{\sum[s(u^*, u, t) \cdot \hat{R}^{(u^*)}(u, t)]}{\sum s(u^*, u, t)}. \tag{3.4}$$

Additionally, this summation is only over combinations of u and t such that

$$s(u^*, u, t) \geq \alpha \cdot s_{max} \tag{3.5}$$

with $0 \leq \alpha \leq 1$ and s_{max} as the maximum obtained similarity for a given test instance. This means that any RUL whose similarity was below this cutoff would not be used in the estimation.

The parameters λ, α, along with the maximum allowable time lag t_{max} between two curves would be parameters to be configured based on results of a validation set.

3.2 Data Gathering — Feature Creation

The data gathering stage of the work pertains to both obtaining the raw sensor data of the many flights of each aircraft via the AGS software, and the transformation of said data into usable information for the algorithm.

The raw data consists of sensor recordings of all flights from 13 different aircraft, concerning the time period of around 4 years. For each flight, a file containing the evolution over time of over 70 variables is stored. This results in over 72 000 files worth of data, each pertaining to the entirety of the corresponding flight, making them a considerable volume of data, and quite unfeasible to use ‘as-is’.

As such, the first step into making this information serviceable was to discard unnecessary data — in this case, cutout unwanted flight phases from the files. This was possible since the AGS software is able to calculate, based on different factors, the flight phase of each moment of the flight, and represent it via the following numerical mapping, presented in Table 3.1.

Table 3.1: Numerical mapping of the flight phases in the AGS software

Code	Description
0	Illegal
1	Engine Stopped
2	Taxi Out
3	Take-off
4	Rejected Take-off
5	2nd Segment
6	Initial Climb
7	Climb
8	Cruise
9	Descent
10	Approach
11	Final Approach
12	Landing
13	Go Around
14	Taxi In

Seeing as the subject of interest of this work is the control system of the aircraft, the least significant flight phases are those pertaining to when the aircraft is stopped. Hence, any information relating to flight phases 0 and 1 was discarded. While this constituted a considerable reduction on the volume of data, working with flight files was still later found to not be viable: the algorithm would come to require an evolution, from healthy state to failure, for each aircraft, which would involve setting up the flight data back-to-back. This would not only be extremely computationally expensive, given the $72\,000$ files, but it would also likely fail to capture the important characteristics of each flight, degradation-wise. In other words, given that the majority of available variables concern the pilots' inputs in the various controls, and the respective outputs on the control surfaces, this alone yields little information on the degradation state of the control system at that flight.

Given this, the second step into setting up the required information was to create a summary of each flight, containing its most important statistics. That is to say, for each subsystem of the aircraft's control system, a set of features thought to relate to degradation was created. Then, all of these flight summaries were organised in run-to-failure sequences for each aircraft, so as to be studied and used in the algorithm.

Following is the detailing of the creation of these features per subsystem.

3.2.1 Flight Controls Electrical System

By inspection of the many flights of the various aircraft, it was possible to conclude that, for each subsystem with dedicated ACEs, the number of engaged ACE channels at a time is mostly the same, and exceptions seem to happen very sporadically and many times at the very beginning of the flight (in the taxi-out phase). While this may be simply due to the AGS software's mis-categorisation of the flight phase (i.e. maybe the taxi-out phase actually started some seconds before or after the software's acknowledgement), it could also be due to a delay in engaging the ACEs. Moreover, when these exceptions did not happen in the taxi-out phase it could be due to some other problem in the electrical system. So the features created concerning this are variables which count how many times the number of engaged ACE channels in each subsystem is not the respective usual value:

- FSACE: Number of times that the number of engaged FS-ACE channels is not 6 in a flight (starting from the Taxi-Out flight phase);
- HSACE: Number of times that the number of engaged HS-ACE channels is not 1;
- ELACE: Number of times that the number of engaged P-ACE elevator channels is not 4;
- RDACE: Number of times that the number of engaged P-ACE rudder channels is not 2.

Furthermore, with the exception of the Flap/Slat subsystem where all the possible FS-ACE channels are engaged at the same time, there are spare channels in every subsystem. So, for each flight, only a portion of the total number of ACE channels is used, and upon inspection of all flights it was possible to conclude that the groups of engaged/disengaged ACE channels are typically well defined and are interchanged each day. To exemplify, there are eight FCM channels for the elevator P-ACEs, corresponding to 1A, 1B, 2A, 2B, 3A, 3B, 4A and 4B, however only four are used each flight, and there are only two possible combinations: in a single flight, either channels 1A, 2B, 3A and 4B are used, or channels 1B, 2A, 3B and 4A, with exceptions happening extremely sporadically.

Additionally, it was found that the accuracy of some sensor readings depends on which group of channels is used for each subsystem, and because of this, it was thought useful to know which channels are most active in each flight:

- ELACE_ENG: Which group of P-ACE elevator channels is engaged in the flight;
- HSACE_ENG: Which HS-ACE channel is engaged in the flight;
- RDACE_ENG which group of P-ACE rudder channels is engaged in the flight.

3.2.2 Ailerons

The variables concerning the ailerons given by the AGS software are listed in Table 3.2.

Table 3.2: Parameters provided by the AGS software pertaining to the Aileron Subsystem

Parameter	Description
AILL	Aileron Surface Left Position
AILR	Aileron Surface Right Position
ROLL_CW_1A, 1B, 3A, 3B, 4A, 4B	Control Wheel Position FCM 1A, 1B, 3A, 3B, 4A, 4B
ROLL_FC_P1, P2	Control Wheel Force Pilot Sensor 1, 2
ROLL_FC_C1, C2	Control Wheel Force Copilot Sensor 1, 2

From here, it is visible that the first three rows of the table are variables which are basically the input of the pilots (ROLL_CW_xx) and the output on the ailerons (AILL and AILR). In this sense, it was thought useful to somehow evaluate the input-versus-output relationship throughout the flights and see how that might correlate to the degradation of the Aileron Subsystem.

The producer of the aircraft at hand, Embraer, provides maintenance manuals, where Table 3.3 is made available, concerning the ailerons and the control yoke:

Table 3.3: Control yoke rotation *versus* left and right aileron deflection [12]

Control Yoke	Left Aileron	Right Aileron
40^o (LEFT)	25^o (UP)	15^o (DOWN)
35^o (LEFT)	20.64^o (UP)	13.54^o (DOWN)
30^o (LEFT)	17.01^o (UP)	11.84^o (DOWN)
25^o (LEFT)	13.59^o (UP)	10.06^o (DOWN)
20^o (LEFT)	10.52^o (UP)	8.20^o (DOWN)
15^o (LEFT)	7.62^o (UP)	6.27^o (DOWN)
10^o (LEFT)	4.92^o (UP)	4.27^o (DOWN)
5^o (LEFT)	2.36^o (UP)	2.18^o (DOWN)
0^o	0^o	0^o
5^o (RIGHT)	2.24^o (DOWN)	2.29^o (UP)
10^o (RIGHT)	4.36^o (DOWN)	4.74^o (UP)
15^o (RIGHT)	6.43^o (DOWN)	7.33^o (UP)
20^o (RIGHT)	8.33^o (DOWN)	10.15^o (UP)
25^o (RIGHT)	10.21^o (DOWN)	13.22^o (UP)
30^o (RIGHT)	11.96^o (DOWN)	16.63^o (UP)
35^o (RIGHT)	13.67^o (DOWN)	20.51^o (UP)
40^o (RIGHT)	15^o (DOWN)	25^o (UP)

With this information, it was possible to fit functions between the control wheel rotation and the deflections on each of the ailerons, in order to interpolate the data. From here, the inputs for each instant of the flights could be used to compute a theoretical output based on those functions, and then this value could be compared to the actual output on the ailerons. The average difference between the two values throughout each flight was then stored as a flight summary feature, as were the maximum and minimum occurring differences in one flight.

It is of note that this theme involving the comparison between a theoretical output and an actual one is present in the feature creation amongst most flight control subsystems, and furthermore there are many features which represent the same thing, simply for a different subsystem, so a feature naming convention was adopted with common representations across subsystems, which are presented in Table 3.4.

Table 3.4: Feature naming convention commonalities for created features

Representation	Description
AV	Average
MIN	Minimum
MAX	Maximum
DIF	Difference
VAR	Variance
L	Left
R	Right
POS	Position
FC	Force
S	Surface Sensor

So the variables AIL_AVDIF_L (R), AIL_MINDIF_L (R), and AIL_MAXDIF_L (R) represent the average, minimum and maximum occurring differences between expected and actual deflections for the left (right) aileron in a flight, respectively.

On the subject of the Aileron subsystem, it is of value to note that, while there are six variables for the control wheel position, this arises simply because there are six sensor readings total. There are two torque tubes below the cockpit, under the control column, each torque tube having three bell cranks connected to position sensors. As such, the average of the six variables was taken as the pilot input. Furthermore, in an effort to try and also study the possible correlation between the degradation of the subsystem and the level of disparity of all six position readings, the average, maximum and minimum variance in these for each flight were also stored as features in the flight summary, and were called ROLL_CW_AVVAR, ROLL_CW_MAXVAR, and ROLL_CW_MINVAR, respectively.

Regarding the force variables, features were created to try to see if there might be a correlation between possible looseness or stiffness of the control wheel (even if only noticeable in the data) and system degradation: the average, minimum, and maximum absolute forces per flight were then recorded, and named AIL_AVFC, AIL_MINFC, and AIL_MAXFC respectively.

3.2.3 Elevator

Concerning the elevator, the AGS software provides the flight parameters listed in Table 3.5.

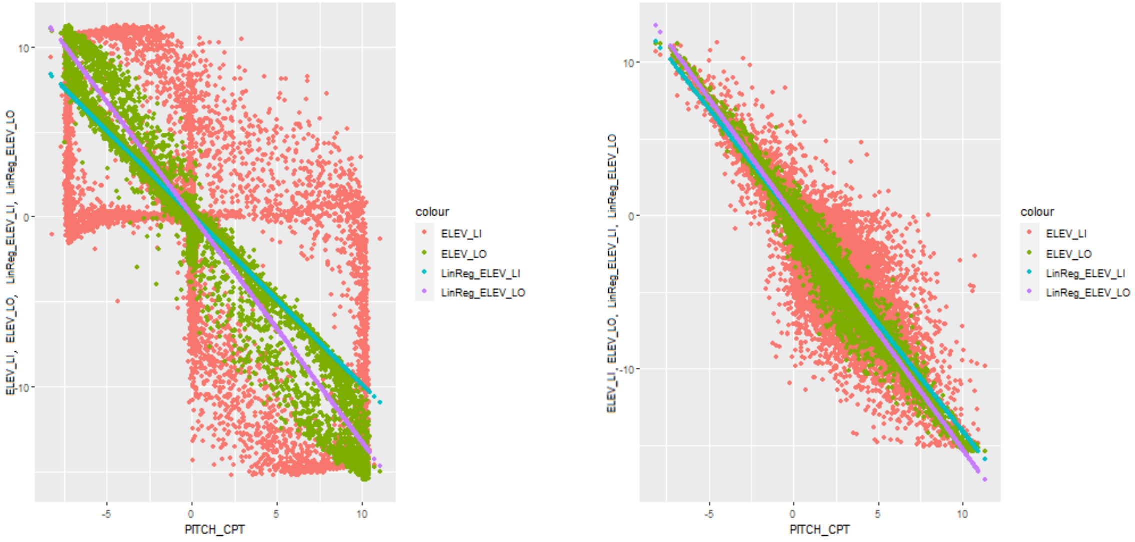
Unlike the ailerons, there is no theoretical input/output table available for the elevator. As such, it is not easy to compute an expected, theoretical, value of deflection based on pilot input, however it is not impossible. Rather than a theoretical relation from the manufacturer, it was instead possible to analyse various flights to obtain a linear regression model for the expected elevator deflection.

Table 3.5: Parameters provided by the AGS software pertaining to the Elevator Subsystem

Parameter	Description
ELEV_LI, LO	Elevator Surface Position Left Inboard, Left Outboard Sensors
ELEV_RI, RO	Elevator Surface Position Right Inboard, Right Outboard Sensors
PITCH_CPT, FO	Control Column Position Pilot, Copilot
PITCH_FC_P, C	Control Column Force Pilot, Copilot

In this case, seeing as the intention was to capture a degradation trend, it was most logical to try and obtain this regression only for flights where the system was deemed “healthy”. Because of this, the lifetime of each aircraft was divided into its many run-to-failure sub-sequences (this task is described in Section 3.3.2 - Lifecycle Splitting. Within these sequences, only a number of initial flights were taken to calculate the regression between the input PITCH_CPT and the various surface position variables presented in Table 3.5.

Furthermore, when plotting the elevator deflection over the pilot input, it was found that the plots differed significantly depending on the flight phase. Among other possible reasons, this is due how slow or how fast the pilots move the controls, as can be shown in Fig. 3.2, which represents two extremes.



(a)

(b)

Figure 3.2: Plots of the left elevator deflection sensor readings (inboard in pink and outboard in green) over the pilot input, for the taxi-out phase (left) and the landing phase (right) of many flights for a specific aircraft

During the taxi-out phase, Fig. 3.2a, the pilots test the aircraft controls by moving them and checking the surface response. These movements are much faster and with a greater range than those typical of when the aircraft is flying, so there is a noticeable delay and the response of the surface is a function of both the pilot input and time, hence this simple linear regression model loses its validity. On the other hand, during the landing phase, Fig. 3.2b, the controls are moved much more slowly, and the response can be approximated as a function of only the pilot input. All other flight phases fall somewhere between these two, thus the phase which was chosen to perform the analysis was the landing, as it was the one where the linear model best applied. Moreover, and for the same reason, the outboard sensor was chosen for the analysis, as it presented the best fit across all aircraft.

From these results, the same principle as the ailerons was applied: the input of the pilots throughout each flight in the specified flight phase was used to calculate the theoretical value for elevator deflection and this result was used to compare to the actual values of deflection. Like the Aileron Subsystem once again, the average, maximum and minimum differences between expected and actual elevator deflections in each flight were then stored in the flight summary, for each of the elevators — left and right. These features were named ELEV_AVDIF_L (and R), ELEV_MAXDIF_L (and R), ELEV_MINDIF_L (and R), totalling six features.

Another set of features arose due to the fact that each elevator surface can be controlled by either of two PCUs (inboard or outboard PCU), each having its own position sensor, so there are two position sensors per elevator surface. Both of the sensors are connected to different P-ACEs, which in turn connect to different FCMs, so a possible difference between the two readings could correlate to degradation or a fault within this circuit. As such the average, maximum and minimum differences between the two sensor readings were computed, for each elevator surface, creating the features ELEVS_AVDIF_L (and R), ELEVS_MAXDIF_L (and R), and ELEVS_MINDIF_L (and R), respectively.

Furthermore, there are two available readings for control column position: one for the pilot, and one for the copilot, while both control columns are mechanically connected. This means both sensors should read similar values, so the difference between the two was also studied. Features pertaining to the average difference, and maximum and minimum occurring differences were stored in variables with names PITCH_POS_AVDIF, PITCH_POS_MAXDIF, PITCH_POS_MINDIF, respectively.

Finally, the average, maximum and minimum forces were also taken to study possible looseness or stiffness of the control column. These features were named ELEV_AVFC, ELEV_MINFC, ELEV_MAXFC, respectively.

3.2.4 Rudder

The parameters for the rudder taken from the AGS software are presented in Table 3.6.

Table 3.6: Parameters provided by the AGS software pertaining to the Rudder Subsystem

Parameter	Description
RUD_PDLFC_P1, P2	Rudder Pedal Force Pilot Sensors 1, 2
RUD_PDLFC_C1, C2	Rudder Pedal Force Copilot Sensors 1, 2
RUD_PDLL, PDLR	Rudder Pedal Position Left, Right
RUD_UP	Rudder Upper Surface Position Sensor
RUD_LO	Rudder Surface Position Sensor

Like the elevator, a table regarding pilot input versus control surface deflection is not provided by Embraer, so the procedure was the same. A group of flights where the control system was assumed to be healthy was used to compute an input/output relation for each flight phase, and then all of the flights were assessed based on those relations. The average, maximum and minimum differences for each flight were taken as the flight summary features RUD AVDIF, RUD MAXDIF, and RUD MINDIF, respectively.

Another considered set of features relates to the fact that the rudder surface can be operated by either the upper or lower PCU, each having its own position sensor, all the while the rudder being considered a single rigid body, for which, in theory, the two sensors should read similar values. For that reason, the average, maximum, and minimum differences between the two readings throughout the flights were also taken as features: RUDS_AVDIF, RUDS_MAXDIF, RUDS_MINDIF, respectively.

There are also two values for the position of the rudder pedals and once again it was thought that the evolving difference between the two readings might correlate to degradation, so the average, maximum and minimum difference between them was taken per flight: RUD_PDL_AVDIF, RUD_PDL_MAXDIF, and RUD_PDL_MINDIF, respectively.

Finally, features pertaining to the force exerted on the pedals were also taken. The average, maximum, and minimum force for each flight were stored in variables RUD_PDL_AVFC, RUD_PDL_MAXFC, and RUD_PDL_MINFC, respectively.

3.2.5 Horizontal Stabiliser

There were no meaningful variables directly pertaining to the Horizontal Stabiliser available in the flight files, however the vast majority of Horizontal Stabiliser faults in the dataset were either classified as false positives (see Section 3.3.2 - Lifecycle Splitting) or have their source in the Electrical System, so it was understood that the lack of features directly representative of this subsystem was not problematic.

3.2.6 Flaps

For the flaps, the AGS software provides the parameters listed in Table 3.7:

Table 3.7: Parameters provided by the AGS software pertaining to the Flap Subsystem

Parameter	Description
FLAP_LVR	Flap/Slat Lever Position
FLAPC	Flap Surface Corrected Position Angle

Additionally, Embraer makes the following table available in their maintenance manuals, concerning the position of the flap lever versus the flap surface position, Table 3.8:

Table 3.8: Flap/Slat lever position *versus* flap deflection in degrees [12]

Flap/Slat Lever Position	Inboard Flap Position (°)	Outboard Flap Position (°)
0 (UP)	0	0
1	7.1	7.0
2	10.1	10.1
3	20.2	20.0
4	20.2	20.0
5	20.2	20.0
FULL (DOWN)	37.1	36.5

Here, both the input and the output are considered discrete variables: the input because there are only 7 possible positions for the lever, and the output because, despite there being a continuous transition from an angle to another on the flaps, the AGS software rounds any value to its closest entry in this table. In fact, the software further rounds all entries to the unit, so a table regarding the FLAP_LVR and FLAPC variables would rather look like Table 3.9:

Table 3.9: Flap/Slat lever position versus flap deflection with values as read in the AGS software

FLAP_LVR Value	Expected FLAPC Value
0	0
1	7
2	10
3	20
4	20
5	20
6	37

where the second column pertains to expected values because there is always a delay between engaging the flap/slat lever and the flap surfaces arriving at their correct deflection.

This meant that creating features accounting for the difference between theoretical and actual flap deflection like the other surfaces would simply be a very indirect way of measuring the response delay, seeing as the surface is always reported by the software to eventually arrive at the exact theoretical deflection. Because of this, the features which were created were aimed at assessing the response time of the surfaces more directly. A first approach was to simply measure the delay every time the lever was moved and take the average, maximum, and minimum delays per flight, which however had the drawback of not taking into account that for example going from a deflection of 20^o to 37^o naturally takes longer than going from 7^o to 10^o, as the system does not adjust the speed based on target deflections. So, instead the features which were taken concerned the mean angular velocity of the flaps: every time the lever was moved, the initial and final deflections and the measured delay between them were used to compute this velocity, and then a mean, maximum and minimum velocity would be taken per flight.

Moreover, by inspection of the flights, it was found that the velocities when increasing the flap deflection were considerably different from velocities when decreasing it, so these cases were treated separately and different features were created for each. So, features accounting for the average, maximum and minimum flap extent speeds were created, and named FLAP_AV_EXTSDP, FLAP_MAX_EXTSPD and FLAP_MIN_EXTSPD respectively, along with features accounting for the average, maximum and minimum flap retract speeds, named FLAP_AV_RETSDP, FLAP_MAX_RETSPD and FLAP_MIN_RETSPD respectively.

However, characteristics of certain flights made it so that these features were very susceptible to missing values. This issue is discussed in Section 3.3.3 - Feature Selection/Feature Plot Visual Inspection, and its implication was that it was necessary to create an additional feature for the flaps: the fraction of time in a flight where the flaps are in the wrong deflection given lever input and considering an ideal response of the surfaces without any delay (moreover, the flight phases corresponding to cruise and climb were not used for this calculation, as flaps are not used there). This feature was named FLAP_TIMEFRACTION.

3.2.7 Slats

The following parameters, listed in Table 3.10, are those made available by the AGS software regarding the slats:

Table 3.10: Parameters provided by the AGS software pertaining to the Slat Subsystem

Parameter	Description
FLAP_LVR	Flap/Slat Lever Position
SLATC	Slat Surface Corrected Position Angle

Furthermore, like the flaps, Embraer provides a table correlating the Flap/Slat lever position and the slat surface angle, as shown in Table 3.11.

Table 3.11: Flap/Slat lever position *versus* slat deflection [12]

Flap/Slat Lever Position	**Slat 1 Position** (°)	**Slat 2, 3 and 4 Position** (°)
0 (UP)	0	0
1	12	15
2	12	15
3	12	15
4	20	25
5	20	25
FULL (DOWN)	20	25

The slats were therefore treated the same as the flaps, as the same concepts apply: the mean, maximum and minimum slat extend speeds were taken (SLAT_AV_EXTSDP, SLAT_MAX_EXTSPD and SLAT_MIN_EXTSPD respectively), along with the mean, maximum and minimum slat retract speeds (SLAT_AV_RETSDP, SLAT_MAX_RETSPD and SLAT_MIN_RETSPD respectively).

Yet once again like the Flaps, a feature had to be created due to the others being susceptible to missing values, accounting for the fraction of time in a flight where the slats are considered to be in the wrong deflection, with the correct deflection being an ideal Slat system with no input/output delay. This feature was called SLAT_TIMEFRACTION.

3.2.8 Spoilers and Speed Brakes

The available sensor readings concerning the Spoiler subsystem follow in Table 3.12:

Table 3.12: Parameters provided by the AGS software pertaining to the Spoiler subsystem

Parameter	**Description**
SPD_BRK	Speed Brake Lever
SPD_BRK1, 3, 4	Speed Brake Lever Position FCM 1A, 3A, 4A
SPOIL_INNL, R	Multi-function Spoiler Surface Position Inner Left, Right
SPOIL_MIDL, R	Multi-function Spoiler Surface Position Middle Left, Right
SPOIL_OUTL, R	Multi-function Spoiler Surface Position Outer Left, Right

where the SPD_BRK variable informs on the source of the commands for the Spoilers at that instant — autopilot or the speed brake lever.

Like the Elevator and Rudder subsystems, the means to compare theoretical deflection to actual deflection throughout the flights had to be done via a linear regression on flights assumed not anomalous to compute a theoretical input/output model. From this, and given lever position throughout the flight when the autopilot is not the agent, the expected deflections were computed, and the average, maximum and minimum differences between these values and the actual recorded ones were stored as features named SPOIL_AVDIF, SPOIL_MAXDIF and SPOIL_MINDIF, respectively.

Moreover, given the three different readings for the speed brake lever position, the possible correlation between faults and the level of disparity between them was studied via the features for average, maximum, and minimum variance on these readings: SPD_BRK_AVVAR, SPD_BRK_MAXVAR and SPD_BRK_MINVAR, respectively.

Additionally, and seeing as when spoiler movement is given by the lever all of the spoilers should move to the same positions, features regarding the disparity between all of the readings were also taken, which is to say the average, maximum and minimum variance per flight: SPOILS_AVVAR, SPOILS_MAXVAR, and SPOILS_MINVAR, respectively.

3.2.9 FHDB Messages

In addition to having data pertaining to the various sensor readings on the flight control system, a database containing all the maintenance messages that the system shows was also available. From this data base, it was possible to extract all the messages related to the ATA Chapter 27, along with the amount of times they were active per day, for each aircraft. In total, there are 603 different possible messages. The objective was to create a feature for each message for the flight summary, whose values would correspond to the message count in that flight. The two major problems with this were that there would be too many features, and that it was not possible to know the message count per flight, only per day. The latter issue was circumvented by assuming that in every flight the message count was the average count between all flights of that day. As for the other issue, many steps were taken into reducing the number of features pertaining to these messages.

First, any messages which never showed in any aircraft throughout the ∼4 year period were discarded, which reduced their number to 285. After this, all the messages were organised as features in the flight summaries. Because this data is organised on a per-flight basis, any messages that showed in days where there were no flights for that aircraft do not show in the data. This further decreased the number of possible messages to 239.

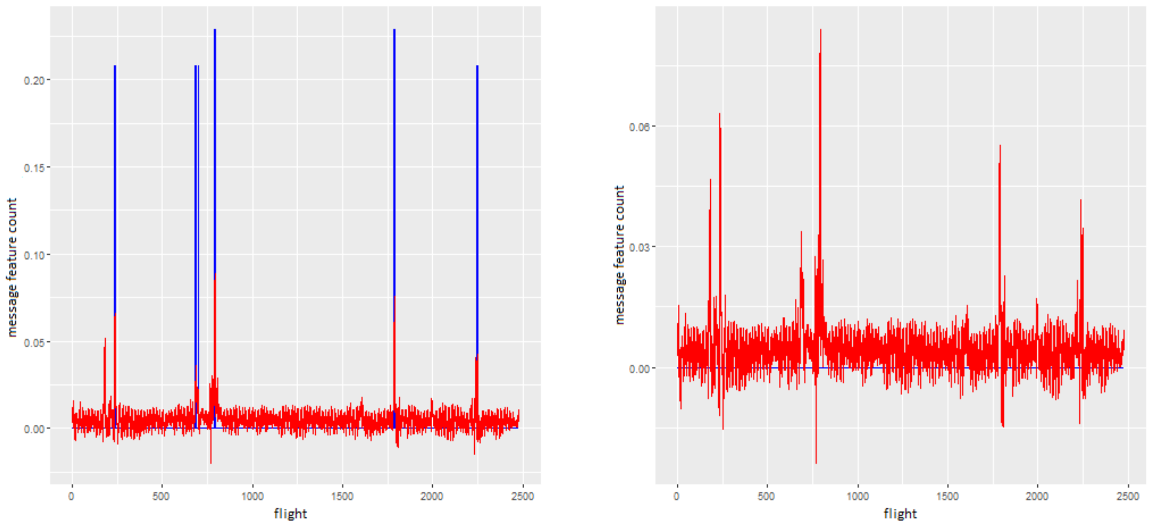
Following this, it was found that there were many messages which could be grouped as a single message, because they pertained to the same event, simply a different FCM channel, or something akin (e.g., "PROC1(MW1)/PROC4(MW2)[FCMxx] MCMPR", where "xx" can be either 1A, 1B, 2A, 2B, 3A, 3B, 4A or 4B). In these cases, the feature which was taken was the sum of the counts of all of these similar messages, which decreased the feature number to 117.

From this step, it was deemed useful to remove messages with a low count, as even if there might be a strong correlation to degradation, the variables were so sparse that they were unlikely to influence the algorithm in any meaningful way. This sparseness can be further appreciated when considered that when in a flight a message is activated by the system, often it is more than once, so for example a total count of 20 does not mean that the message appeared in 20 separate days, rather it appeared more times in less days. Given all of this, the cutoff was set to 100: if the message count was a total of 100 or less throughout the 4 year period and throughout all aircraft, it was discarded, which resulted in 60 total messages left.

A correlation matrix was then computed to try to reduce pair-wise correlation in this dataset. Resulting from this step, the number of total messages decreased to 46.

Next, messages pertaining to subsystems that were thought to not be as important to gauge the overall system's health were also discarded. In this case, it was found that there were many messages for the Slat/Flap subsystem, which had only been the direct cause of a failure event in the dataset once for all aircraft. This further reduced the message count to 30.

However, in later testing it was found that the majority of these remaining features were still too sparse. This issue stemmed from the fact that, for most of these messages, the features from data for training the autoencoder (data which had been deemed to belong to healthy periods of the aircraft's lifetime) were mostly always zero. In the reconstruction phase of the algorithm, where the data is no longer necessarily healthy, the message count is no longer always zero, however, by this time, the autoencoder has already learned that, not only are all these features supposed to be zero, but they are also "supposed" to be all equal to each other (as the autoencoder learns the features as well as their relation to each other). This caused rather undesirable results in the reconstruction of the features whenever one of the counts for a message presented a relatively high value, as the autoencoder tried to reconstruct that feature, but then tried to add the same characteristics of that plot into features from other messages, as shown in Fig. 3.3, where two of the message features are presented, for a single run-to-failure sequence on a specific aircraft.



(a)

(b)

Figure 3.3: Two message features for a specific run-to-failure sequence (blue), along with their respective reconstructions (red); the reconstruction of the feature on the left influences the reconstruction of the feature on the right

This meant that the error which was supposed to arise from one feature only was being amplified by a potential 29 others. Along with the fact that the path towards reducing the number of used messages, explained above, might have led to the discarding of important information, this prompted the dropping of this method in favour of some other.

If the problem was the sparseness of the message features, then a workaround was to sum all of the message counts, which would along solve the high feature number complication. With this objective, it was decided that messages pertaining to each subsystem were to be summed. Moreover, these should be weighted sums, so as to somehow take into account message importance, as there were messages which appeared many times, yet were widely ignored by the maintenance team for not being important, while there were others which seemed to immediately grant intervention. As such, all maintenance reports pertaining to the Flight Control System which mentioned the replacement of a component (these reports are discussed further in Section 3.3.2 - Lifecycle Splitting) were gathered. Then, the number of reports where each message was mentioned was stored.

From this, 5 categories of message importance were created, based on both the number of occurrences in reports, and also the type of fault they were attached to (i.e., if the report was due to a FLT CTRL NO DISPATCH or not). These are shown in Table 3.13 by order of least important to most important.

Table 3.13: FHDB message importance based on mentions in maintenance reports

Category	Condition
I	Message does not appear in reports
II	Message appears less than 5 times in reports
III	Message appears 5 or more times in reports
IV	Message appears less than 4 times in FLT CTRL NO DISPATCH reports
V	Message appears 4 or more times in FLT CTRL NO DISPATCH reports

Weight values between 0 and 1 were then assigned to each category, and the weighted sums per subsystem were finally computed. These were the features used in the model: AIL_MSG, ELEV_MSG, FLAPSLAT_MSG, RUD_MSG, MFS_MSG and ELECSYS_MSG, for the Aileron, Elevator, Flap/Slat, Rudder, Spoilers, and Electrical subsystems sums, respectively.

3.3 Data Pre-Processing and Feature Selection

The data pre-processing stage pertains to the steps taken into preparing the gathered data to be input in the algorithm, such as dealing with abnormalities in the data, rearranging the flight summaries into run-to-failure sequences, selecting which features should be used in the model, normalising the data, and creating the train, test, and validation sets to train, tune, and test the model. These steps are discussed in the following sections.

3.3.1 Abnormalities in the Raw Data

One aspect of the raw data which has not yet been mentioned is the appearance of unexpected abnormalities which are not necessarily associated to faults, yet some could later pollute the gathered features if left unchecked. These abnormalities are addressed in the following paragraphs.

Unusually Long Flights

In rare occurrences, the file corresponding to a certain flight would be so long as to belong to a flight lasting upwards of 100 hours. When inspected, these files also showed odd data, such as the flight starting in the cruise flight phase, going back to taxi-out, and back to cruise multiple times. Flight summaries belonging to any of these flights (any flight lasting over 5 hours as discussed and stipulated with Portugália Airlines given their normal operations) were discarded in order to prevent the appearance of these situations in the data, which do befoul the gathered features, as shown in Fig. 3.4, where for the FSACE feature for a certain aircraft, these unusually long flights create two points on the plot completely dwarfing the rest.

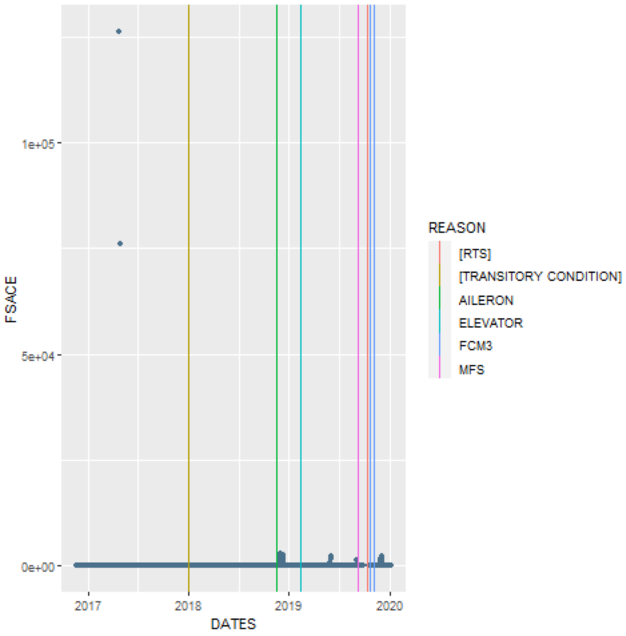


Figure 3.4: Example of the effect of an abnormally long flight on the collected features

Repeated Flights

Another irregularity in the data concerns separate files which supposedly report different flights, yet the aircraft along with the exact date and time are the same. In these cases, the contents of the files, although not exactly the same, are very similar, only differing slightly in length. To deal with this, the flight summary of only the first file was kept (which is to say, the file with the lowest flight ID, as they both carry the same date).

Missing Values

Some of the created features are somewhat prone to missing values, as they require some sort of condition to be true in order to be computed. As will be discussed in Section 3.3.3 - Feature Selection/Feature Plot Visual Inspection, on the features which were not discarded and were kept for inputting in the algorithm, this issue was dealt with by replacing the missing values with the median of the respective feature, as the missing values were found to not correlate to system degradation.

3.3.2 Lifecycle Splitting

As explained before, in the Algorithm Overview section, the data needed to be organised in a very specific way so as to be used by the algorithm. Specifically, after having created the features and systematising them into flight summaries (each flight summary containing the features obtained for a single flight), it was necessary to further organise them by aircraft and by date, but most importantly, to arrange them in run-to-failure sequences.

In order to do this, the flight summaries were ordered by date for each aircraft, and where there was a report of FLT CTRL NO DISPATCH, this main sequence was cut, originating a run-to-failure sequence.

True Positives and False Positives

Despite this task's apparent triviality, various issues arose in its implementation largely due to the faults' trustworthiness. This is to say that the fact that there existed an event of FLT CTRL NO DISPATCH is not an absolute confirmation that there was an actual fault on the system, and this is possible to assess via the reports of the maintenance team — the work-orders.

A work-order that arises due to a FLT CTRL NO DISPATCH message on the CAS typically provides the steps which were taken into dealing with the fault, along with the information the maintenance team used to isolate the fault, if any (i.e., maintenance messages active on the CMC of the aircraft, which provide a fault code that can be cross checked with the aircraft's maintenance manuals). With this information, it is possible to know when a component had to be replaced, or some other corrective action had to be taken. These events were treated as true positives, as the CAS message did not disappear until the action was taken, which indicates that there was indeed a fault in the system.

However, not all work-orders report an action such as described above, and instead they point to failure events which might be more ambiguous. Specifically, those work-orders were labelled such that there are four main categories of maintenance action which can lead to either suspicion that the event was a false positive, or in simpler cases actual confirmation that there was no fault: a Control System Return to Service (RTS) test, Power Down/Power Up (PDPU), Transitory Condition of the system, or simply a maintenance report which is too unclear or does not specify what action was taken. These actions/events are henceforth referred to as RTS, PDPU, TRANSITORY CONDITION, and UNCLEAR, respectively, and are explained in the following paragraphs.

An RTS is, as the name suggests, a test performed by the maintenance personnel to determine whether the control system of the aircraft is fit to fly. If it passes the test, it is allowed to return to service, and if it does not, further action is required. The reality of it is that it is not uncommon for the aircraft to have a FLT CTRL NO DISPATCH message display on the CAS, the RTS being successful, but the message is displayed again at a later flight. Table 3.14 gives three examples of this in the data, where the column named "Reason" pertains to the source of the fault code associated with the fault, which might be shown in the CMC, even in cases involving an RTS. This is the code the maintenance team would have used as a starting point to isolate the fault, had the test not been successful. In these RTS cases, however, no fault code had been given, or at least it was not reported. Table 3.14 shows then that these events might have actually been true positives, and because of this, the trustworthiness of the test may be questioned.

Table 3.14: Examples of successful RTS tests on events of failure that might have been true positives

Timestamp	Reason	Action
11/12/2017 00:00:00	(None given)	RTS
13/12/2017 07:39:36	Rudder	Rudder Surface Rigging Performed
01/03/2018 12:24:00	(None given)	RTS
01/03/2018 13:07:48	Rudder	Rudder Surface Rigging Performed
17/08/2016 12:49:48	(None given)	RTS
21/08/2016 07:19:48	(None given)	RTS
22/08/2016 05:58:48	Rudder	Lower Rudder PCU Replaced

Despite this, many other events where the RTS test was successful may actually be considered false positives with some degree of certainty. The best example of this is shown in Table 3.15, where all faults of a single aircraft are listed. The fact that these events are so far apart, and with none of them having an associated fault code suggests that these were indeed false alarms.

Table 3.15: All FLT CTRL DISPATCH events recorded for a single aircraft

Timestamp	Reason	Action
24/01/2017 00:00:00	(None given)	RTS
13/07/2017 06:00:36	(None given)	RTS
07/02/2019 13:48:00	(None given)	RTS
20/10/2019 13:15:36	(None given)	RTS
31/01/2020 04:42:00	(None given)	RTS

With all of these considerations concerning RTSs, the action which was taken regarding the splitting of the data into run-to-failure sequences was that all of these faults should be ignored. In the cases where there was no given fault code and the fault was isolated, it was considered to be a false positive. On the other hand, in the cases where they were close to other faults, with or without an associated fault code, it was decided that the sequence should only be cut on the fault which had a corrective action taken. This was because flights between the initial RTS and the final fault may still be of use in describing a very degraded system. In fact, the last flight before the fault which was acted on describes a system in such way critically degraded that it no longer passes the RTS test where, before, it did.

A PDPU is a simple powering down of the aircraft, followed by a powering up. If the CAS message disappears from the display, the aircraft is deemed suitable for flight, and no further action is taken. These cases are far more rare than RTSs, and they commonly also lead to the situation that was represented by Table 3.14. Because of this, PDPU events were treated the same as RTS events: they were ignored when creating the run-to-failure sequences.

A TRANSITORY CONDITION pertains to faults where the reason for the emerging of the CAS message was attributed to some transient state in the system. This attribution can also occur when the aircraft showed a fault message, but when it arrived at the hands of the maintenance team, this message was no longer present, and the fault could not be replicated. In these cases, no corrective action is taken. Like the RTS cases, many FLT CTRL NO DISPATCH events labelled TRANSITORY CONDITION precede another event where corrective measures had to be taken, so these cases were treated the same as RTSs, once again.

Finally, UNCLEAR cases label reports that were too unclear or did not specify what action was taken to deal with the fault. In these cases, there was not enough information to assess the trustworthiness of the fault, but since a maintenance action involving the replacement of some part or something akin would likely be clearly reported by the maintenance team, these faults were taken as false positives, and hence ignored.

Very Short Sequences

When splitting the flight summaries into run-to-failure sequences, some would end up very small, as there were faults very close to each other. These small sequences were discarded for the possibility that there was not much of an evolution of degradation, rather the system was never healthy to begin with. The number of flights for this cutoff would later be studied and selected.

Successive Corrective Actions on the Same Component/Subsystem

In addition to the true positive versus false positive issue, special attention was also given to cases where there was a reported fault which was acted on, yet soon after the same fault arose again, indicating that the maintenance intervention either failed to address the original problem (at least in its full amplitude), or originated another instance of the same or similar problem. This may be exemplified by the data presented in Table 3.16, where the faults regarding the multi-function spoilers represent the former problem, while the faults regarding P-ACE 1 stand for the latter.

Table 3.16: Examples of successive corrective actions regarding the same fault

Timestamp	Reason	Action
28/07/2018 06:51:00	Multi-function Spoilers	Multi-function Spoilers Left Inboard PCU Replaced
29/07/2018 04:21:00	Multi-function Spoilers	Swapped FCM 3 with FCM 2
10/10/2016 08:36:36	P-ACE 1	P-ACE 1 Replaced
11/10/2016 02:58:12	P-ACE 1	P-ACE 1 Replaced

The P-ACE 1 faults represent the vast majority of this kind of issue: a component was faulty and was therefore replaced, however its replacement also originated a fault, so corrective measures had to be taken again.

Unlike the cases involving an RTS, PDPU or TRANSITORY CONDITION, here the issue persisted until corrective measures were taken, so it was considered that the run-to-failure sequence should end at the first fault. As for the faults which followed, the corresponding sequences would be very short, so they were discarded as discussed before.

A special case of the same type as described above worth mentioning concerns reports where the fault was not successfully corrected, resulting in the aircraft not being allowed to fly — Aircraft on Ground (AOG). In this scenario, another report is made when the fault is corrected, and for the purpose of this project, these cases were treated the same as described above: the sequence should be cut at the first report, and the following sequence (which here would have no flights) was discarded.

Maintenance Checks

The final obstacle in obtaining the run-to-failure sequences were maintenance checks. Maintenance checks are scheduled maintenance actions which can take up to months to undergo. The problem they present is that each sequence should be a "natural" degradation of the system. If major maintenance work was done on the system, then there likely is a discontinuity in its behaviour which does not correctly portray its normal evolution. This can be shown in the created features, as presented in Fig. 3.5.

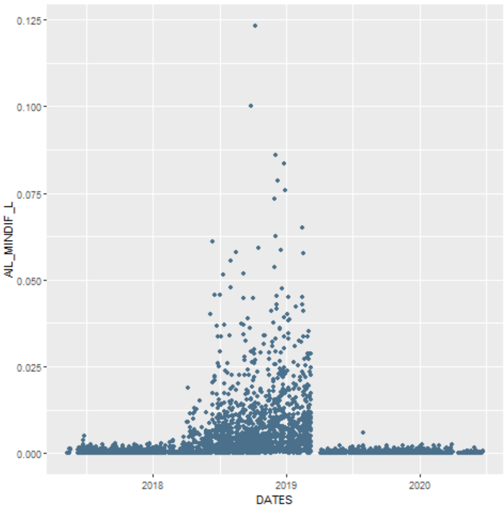
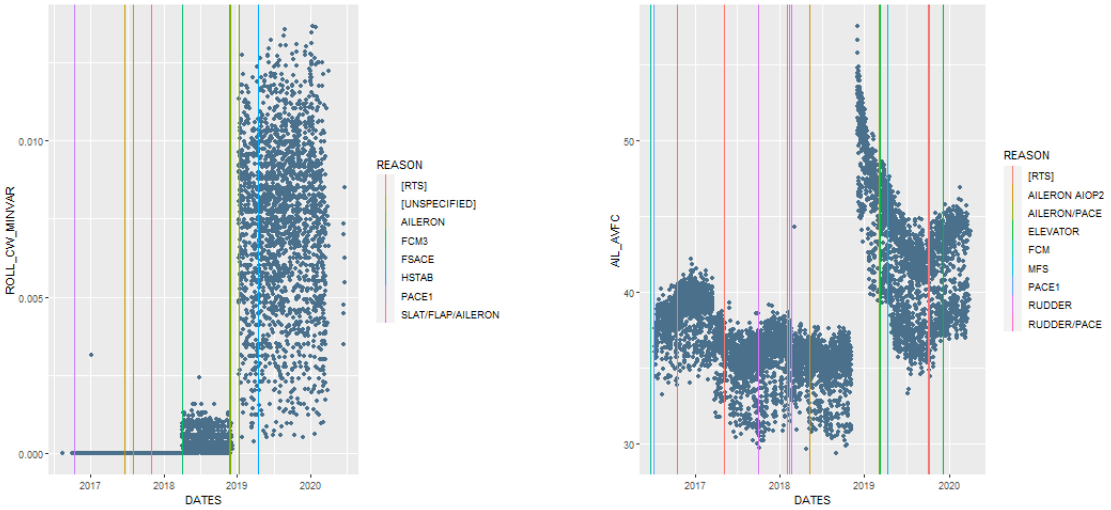


Figure 3.5: Example of changes in behaviour after a maintenance check

There, a period of no flights is visible in the first quarter of 2019, along with the change in behaviour of this particular feature, after this period.

Another posing issue pertains to the fact that it is not unusual that certain features show a behaviour worsening after the maintenance check, as shown in both plots of Fig. 3.6. In these cases, Fig. 3.6a shows a greater disparity between the control wheel position sensors after a check at the end of the year 2018, while for a different aircraft Fig. 3.6b shows that the control wheels became remarkably stiffer, after the check sometime near the end of the year 2018.

Note that here (Fig. 3.6) the most common plot type in this text is introduced. The scatter points represent the value of the respective feature in a certain flight, while the vertical lines represent the dates at which there was a fault. Here, the colour of the lines further denotes the reason of the corresponding fault (the same reason introduced for tables such as Table 3.17).



(a)

(b)

Figure 3.6: Examples of maintenance checks which resulted in not only a change in behaviour, but also a worsening of certain features

Fig. 3.6a shows another difficulty. Right after the maintenance check there was a fault on the Aileron subsystem: was this fault a culmination of the evolution of the subsystem, or was it caused or aggravated by the maintenance check? Inspection of the Aileron faults around this period (as shown in Table 3.17) shows that before the maintenance check, there had already been several Aileron faults, all labelled either RTS, PDPU, TRANSITORY CONDITION, or UNCLEAR, and that the last intervention, which included corrective action, ended this succession of failures.

Table 3.17: Reported aileron faults in the period around the maintenance check

Timestamp	Reason	Action
23/11/2018 12:12:00	(None given)	RTS
24/11/2018 12:40:12	Aileron	PDPU
25/11/2018 05:22:12	Aileron	UNCLEAR
07/01/2019 09:09:00	Aileron	TRANSITORY CONDITION
07/01/2019 09:42:36	Slat/Flap/Aileron	RTS
07/01/2019 09:45:36	Aileron	AIOP 1A and Left Inboard Aileron PCU Replaced

There are then the following possibilities for the source of this fault: either the aileron faults before the maintenance check show that the subsystem was already degraded and it would invariably fail, requiring corrective action at some point, or the maintenance check created or exacerbated the issue, which is not unreasonable to think, given the visible worsening of this feature concerning the aileron.

Given all of this, it was decided that sequences should never encompass periods where a maintenance check was done on the aircraft. If a sequence was to begin before a check and end at a fault after the check, it was discarded. Furthermore, the next sequence should start at the end of the maintenance check. In these cases, one of two things could happen: if there was a fault not too long after the check, it means the system was not healthy right after this maintenance intervention, and the sequence would be so small that it would be rejected along with all other small sequences, as described before; if there was no fault, then the system was considered healthy, so the sequence was used and cut at whichever valid fault came next.

Failure Redefinition

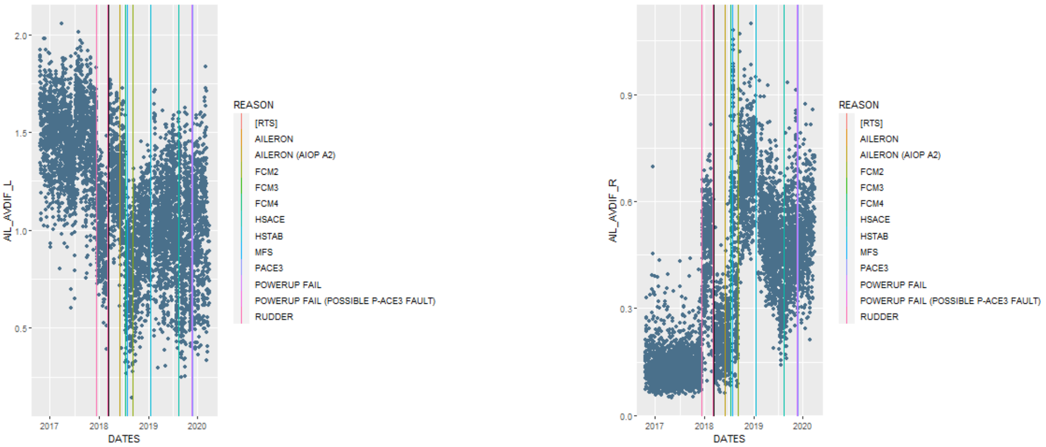
All of these steps to select which faults should be taken into account when splitting the sequences left the data with a problem it already showed before, but was now amplified: lack of failure data. In fact, throughout the almost 4 year period and across 13 aircraft, there were only 179 reports of FLT CTRL NO DISPATCH, and of these a total of 112 pertain to RTSs, PDPUs, TRANSITORY CONDITIONs, and UNCLEARs, leaving only 67 failure events which could, unambiguously, be considered true positives. Moreover discarding certain sequences because of the additional issues described above and considering still that the data had to be divided into train, test and validation sets, the final number of faults was even lower.

While this did not pose a substantial issue towards training the Autoencoder (as the data required for this should pertain to healthy data, of which seemingly there was plenty), it presented a significant obstacle considering that each run-to-failure sequence was going to be used to arrive at a Degradation Curve. As such, each Degradation Curve would show a distinct "path" towards failure, and it was therefore important to have as many sequences as possible so as to be able to compare to curves from a test set.

This originated the need to reevaluate the definition of "failure" which had been used until now — that there was a failure if the CAS displayed the message "FLT CTRL NO DISPATCH" — so all of the work-orders pertaining to the Flight Control System in the considered period were gathered for analysis. This time, attention was given to work-orders which involved corrective action due to a finding of a degraded or faulty component, or a complaint from the pilots — these would be added as extra failure events. This further helped identifying the sources of certain behaviour changes in the features which could otherwise not be explained.

Due to there being close to three thousand reports, it was impossible to individually analyse all of them, and therefore attention was given first and foremost to reports which mentioned the replacement or installation of a component, as these would be the most critical events. After checking these, if there were still unexplained abnormal behaviours or behaviour changes on any features, the rest of the reports was inspected on the periods around those abnormalities.

A common event to report was found to be the replacement of the aileron cables due to them having excessive wear. This usually marks a change in the AIL_AVDIF features. Fig. 3.7 shows plots of these features for a given aircraft.



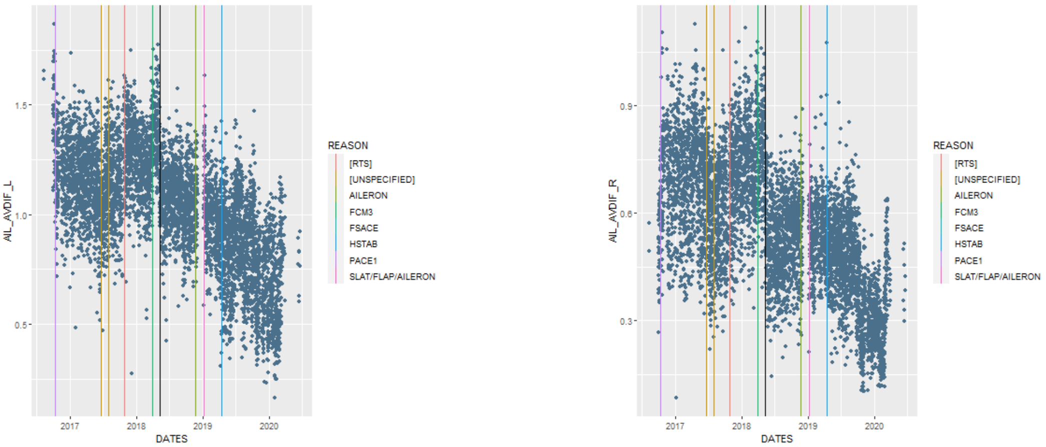
(a)

(b)

Figure 3.7: Example of the effect of aileron cable replacements on two aileron features

The vertical black line marks the first reported date in the studied period at which the maintenance personnel found the Aileron cables with excessive wear, thus replacing them in a certain aircraft. Interestingly, while this intervention seemed to have a positive effect on the right-hand side feature (a lesser difference between theoretical and actual aileron deflection), the same cannot be said for its left-side counterpart, where an increase is visible. Indeed, these features are not only dependant on the state of the cables, but mostly on their calibration, so even small differences on the way they were installed are visible here. This originated difficulties in assessing the components' health via these features, for it was not possible to know in advance whether high values were due to degradation, or calibration. In fact, up to this intervention's date, no degradation trend is visible in these plots, rather only drastic behaviour changes which were more likely to be due to component replacements, or simply calibration actions.

This does not mean this degradation was completely invisible in all aircraft, however, as can be shown in the plots in Fig. 3.8, where an increasing trend is visible (most prominently in the right-hand side feature) in the several months prior to the maintenance action of this same type (as represented by the black vertical line).

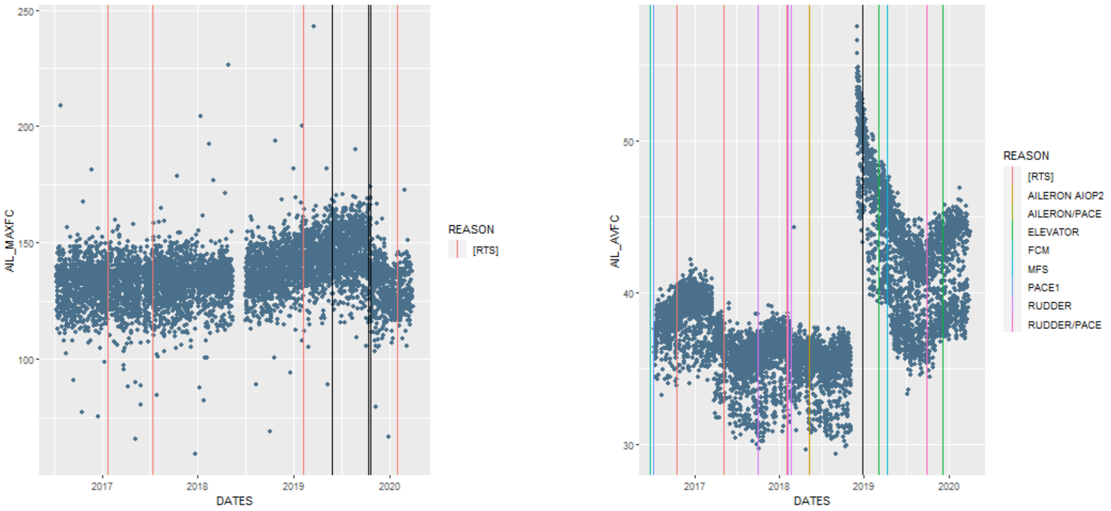


(a)

(b)

Figure 3.8: Example on another aircraft of the effect of aileron cable replacements on two aileron features

Another somewhat common report pertained to the pilots' complaints on the stiffness of the aileron controls. With only one exception (see Fig. 3.9b), these are only well visible on the AIL_MAXFC feature. Fig. 3.9a shows this feature in one of the aircraft, with vertical black lines representing these complaints. The last of these events was the only one where corrective action was taken, as is visible by the abrupt behaviour change in the plot. It is worth noting that despite there being an increasing trend on this plot culminating in the need for maintenance action, the feature's actual values alone in this period would not grant immediate conclusion that there was a problem with the aileron subsystem, as on other aircraft this feature often presents even higher values, without complaints from the pilots.



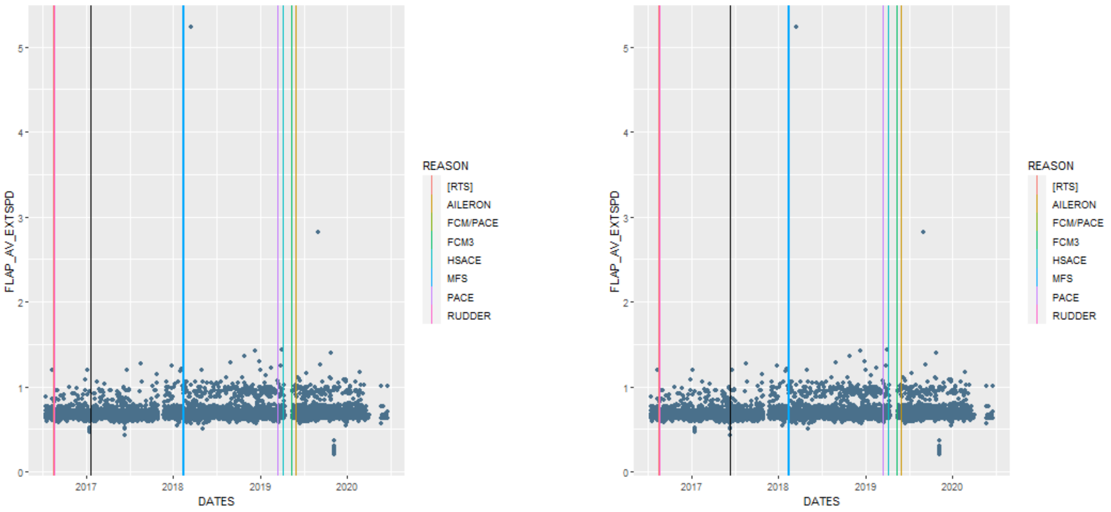
(a)

(b)

Figure 3.9: Examples of aileron force feature correlation to pilot complaints on control stiffness

Fig. 3.9b shows the one exception where these complaints can be cross-checked with not only AIL_MAXFC, but also AIL_AVFC and AIL_MINFC. However, in this instance, no corrective measures were taken, and indeed, unlike the plot in Fig. 3.9a, there is no sudden change in the behaviour of the feature to indicate a maintenance intervention. In these cases, the complaints were not considered faults, as nothing was done to the system, yet it still flew with no further reported issues.

Finally, the most common of reports pertains to issues on either the Flap or Slat subsystems, concerning faults, and consequent replacements, of skew sensors, surface position sensors and actuators. Interestingly, these two subsystems were the least common source of FLT CTRL NO DISPATCH events, and were it not for these new added faults, their respective features would have been discarded. With this new failure data, however, correlations between feature behaviour and faults are visible, as presented in Fig. 3.10, where lower average flap extend speeds are visible in the flights close to the two 2017 flap faults (beginning and middle of the year 2017, as presented by the vertical black lines in Fig. 3.10a and Fig. 3.10b).



(a)

(b)

Figure 3.10: Flap feature correlation to flap faults found in the maintenance reports

These and some other less frequent faults were added to the dataset, and the issue of lack of failure data was reduced.

3.3.3 Feature Selection/Feature Plot Visual Inspection

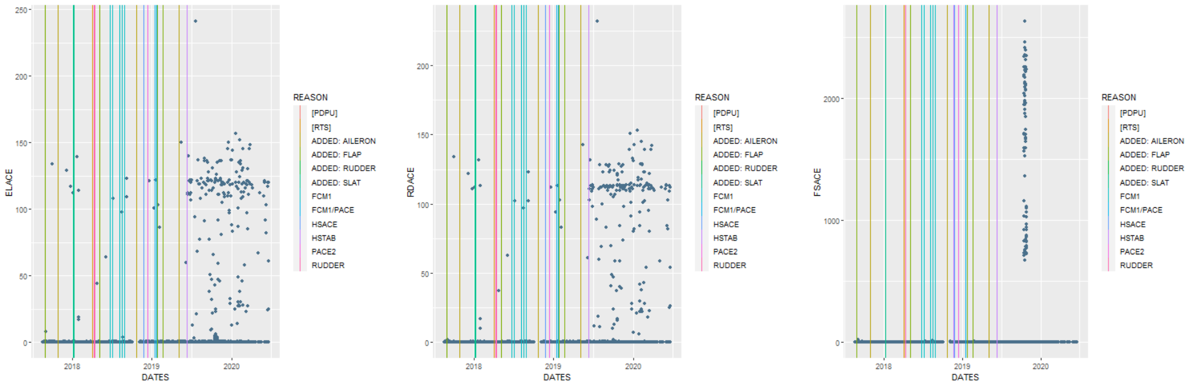
The total created features amounted to 75, yet of these only a selection would later be used in the algorithm, since there were many which would only add to the run time of the algorithm and nothing more. This selection should be made based on correlation to the faults. This was done via a visual inspection of the plots of each feature throughout time (i.e. the dates of the flights) to assess their usefulness in predicting impending faults. This section covers that analysis.

It should be noted that in testing it was found that some sensor readings were susceptible to dramatic outliers both due to the rapid command movements in the taxi-out phase, and the inherent noise the signal carries. In these cases, the respective features which represented an average were found to be more useful when they were used as a median, so they were changed.

The following covers only those plots which were deemed of most interest to show, yet all the plots along with their reasoning for keeping/discarding are presented in Appendix B.

Electrical System

Following in Fig. 3.11 are the plots for features ELACE, RDACE and FSACE for the aircraft chosen to best demonstrate the statements made in the following paragraphs regarding the Electrical System.



(a) (b) (c)

Figure 3.11: Plots for the ELACE (left), RDACE (centre), and FSACE (right) features on a single aircraft

ELACE is, for most aircraft, very sparse, presenting mostly values of zero, and with outliers which do not seem to have any correlation to the faults. In fact, for aircraft where this feature is not so sparse, the distributions of the points seem to bear little to no information about fault imminence. As presented in Fig. 3.11a, there start being more ELACE occurrences mid 2019, and this behaviour continues for at least a year, with no reported issues, which seems to indicate this is not a good candidate for prognostics analysis.

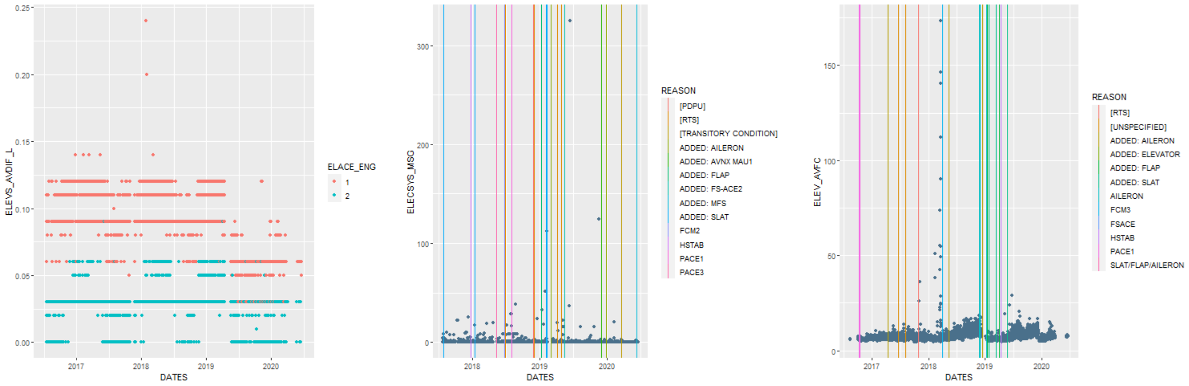
RDACE mostly presents a distribution similar to ELACE, which is not unexpected given that both of these features pertain to P-ACEs. As such, this feature carries the same characteristics as ELACE.

FSACE, like ELACE, is very sparse and its occurrences seem to have no correlation to faults on either the Electrical, Flap, or Slat subsystems. However, and as shown in Fig. 3.11c, it carries a characteristic which will henceforth be referred to as “system-wide peaks”. In this figure, one of these very distinct peaks is visible in the last quarter of 2019 — these are phenomena of anomalous behaviour which, when present at a given period, show in many other features, across multiple subsystems. These abnormalities will be discussed later in this section when analysing the features directly pertaining to the Flap and Slat subsystems.

HSACE was found to always be a constant of zero, therefore carrying no information for the algorithm.

All of these characteristics dictated that these features be discarded, as they were not worth the additional computing time required for having them.

As for the ELACE_ENG, HSACE_ENG, and RDACE_ENG features, their initial function was not to point to system degradation (as they only show which groups of ACEs were chosen for the specific flights), but rather to aid the algorithm in identifying and better dealing with changes in behaviour which are not necessarily due to an impending fault, and instead are due to this change in ACEs, as is shown in Fig. 3.12a, where the ELACE_ENG value is represented by the colour of each point in the plot. This would be advantageous since the measurement of health will be taken from the error of the AE in reconstructing the data it is given. Of these, only ELACE_ENG was kept, as RDACE_ENG largely coincides with it, and HSACE_ENG pertains to the horizontal stabiliser, for which no features were created.



(a) Example of feature dependency on the ELACE_ENG variable

(b) Plot for the ELECSYS_MSG feature for a given aircraft

(c) Example of a fault on the Electrical System's influence on features belonging to other subsystems

Figure 3.12: Plots representing some characteristics of the Electrical System

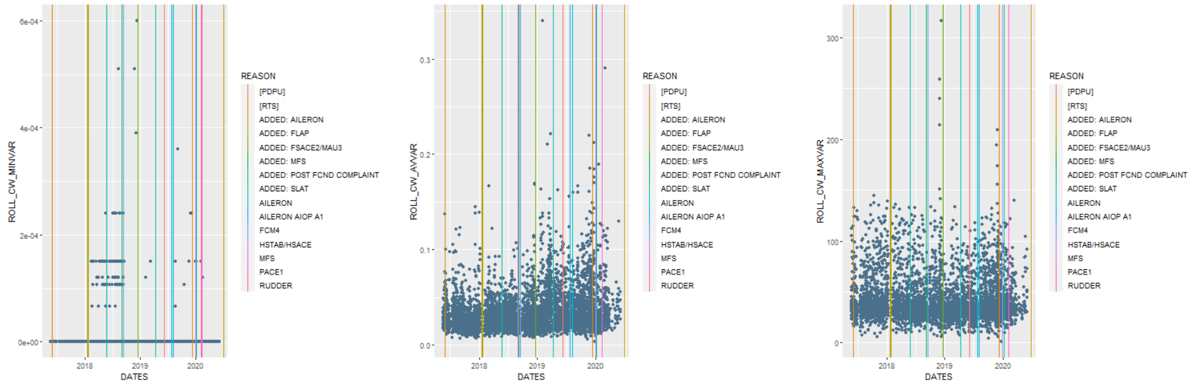
Finally, the ELECSYS_MSG feature shows that its distribution might be related to certain faults not limited to the electrical system. In Fig. 3.12b, for example, the relatively high incidence of outliers around the first half of 2019 is concurrent with the high fault frequency in the same period. In fact, apart from two other outliers, this period ends with a fault, and respective maintenance action, on MAU1.

Given these examples across many aircraft, this feature was kept. It should be noted that it was decided all features regarding the FHDB messages were to be kept, given the importance of these messages and how many of them dictated the actions of the maintenance team.

Furthermore, while this means that there was only one feature directly representative of the Electrical System's degradation state, this should not be taken as a sign of its unimportance. In fact, the most common source of faults in the dataset is precisely this subsystem, so it should be handled as such — and despite this apparent issue, it is in truth not being belittled. Verily, being as global a system as it is, it affects all others, and hence many anomalies in the behaviours of features belonging to other systems (such as the Rudder, or Elevators) can be traced back to faults in the Electrical System. An example of this is given by Fig. 3.12c, in the first quarter of 2018, where an unusual amount of force seems to be needed to move the elevator control column in some flights; this issue eventually ends after a fault and its respective corrective action on FCM1 — an Electrical System component. As a final note on this matter, although it might seem so, the described behaviour in Fig. 3.12c is not a system-wide peak, as it is exclusive to this feature at this period for the given aircraft.

Ailerons

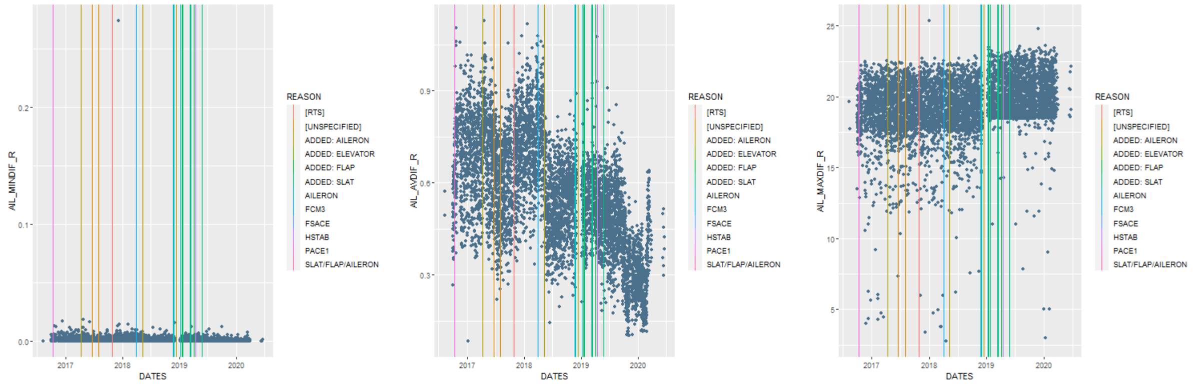
Of the three ROLL_CW_xxVAR features (presented in Fig. 3.13 for a specific aircraft), only ROLL_CW_MINVAR was kept, for two reasons. The first being its clarity in possible fault correlation, which is visible in Fig. 3.13a as a behaviour change by the second half of 2018 extending to the first quarter of 2019, which worsens by the end of 2018, and eventually ends at an FCM4 fault. The second reason was for this feature generally being, of the three, the one least affected by system-wide peaks (as will be shown later in this section that these peaks come in detriment to the data, rather than improvement).



(a) (b) (c)

Figure 3.13: Plots for the ROLL_CW_MINVAR (left), ROLL_CW_AVVAR (centre), and ROLL_CW_MAXVAR (right) features on a single aircraft

Regarding the set of features presented in Fig. 3.14, along with their left side counterparts, only AIL_AVDIF_R and AIL_AVDIF_L were kept, seeing as they presented the greatest variability with which coming visible behaviour changes after maintenance actions. Fault correlation with these features has already been discussed in the paragraphs accompanying Fig. 3.7 and Fig. 3.8, in Section 3.3.2 - Lifecycle Splitting so this is not repeated here. An additional reason these features were chosen over the rest was their imperviousness to system-wide peaks.

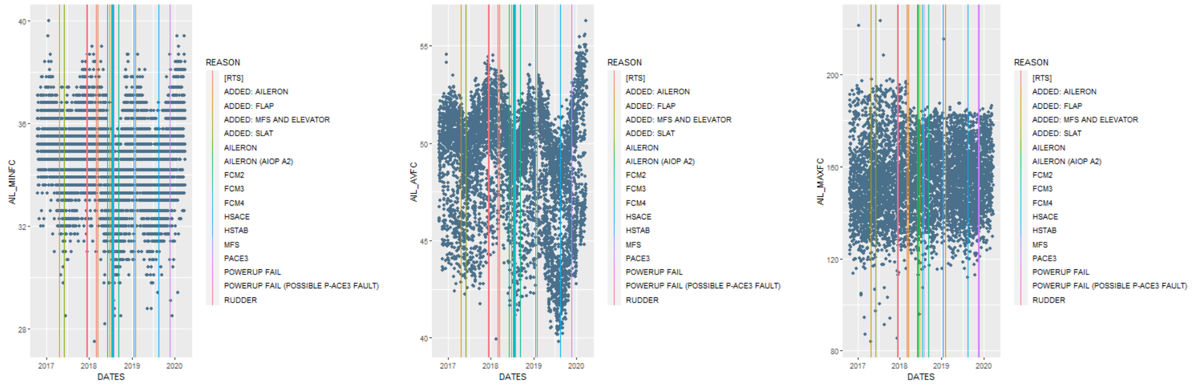


(a) (b) (c)

Figure 3.14: Plots for the AIL_MINDIF_R (left), AIL_AVDIF_R (centre), and AIL_MAXDIF_R (right) features on a single aircraft

Finally, for the aileron force features (Fig. 3.15), both AIL_MINFC and AIL_AVFC display seasonal behaviours which are only disrupted by maintenance actions on the system (see Fig. 3.9b for an example of such disruptions). According to these two features, generally, more force is required to move the aileron controls in the winter season, and conversely, less is required during the summer season.

As for AIL_MAXFC, it is the only aileron force feature independent from season, and it is also in fact the one where the majority of complaints on control stiffness is visible, as discussed before, in Section 3.3.2 - Lifecycle Splitting. For this reason, it was the only one which was kept.



(a) (b) (c)

Figure 3.15: Plots for the AIL_MINFC (left), AIL_AVFC (centre), and AIL_MAXFC (right) features on a single aircraft

Elevators

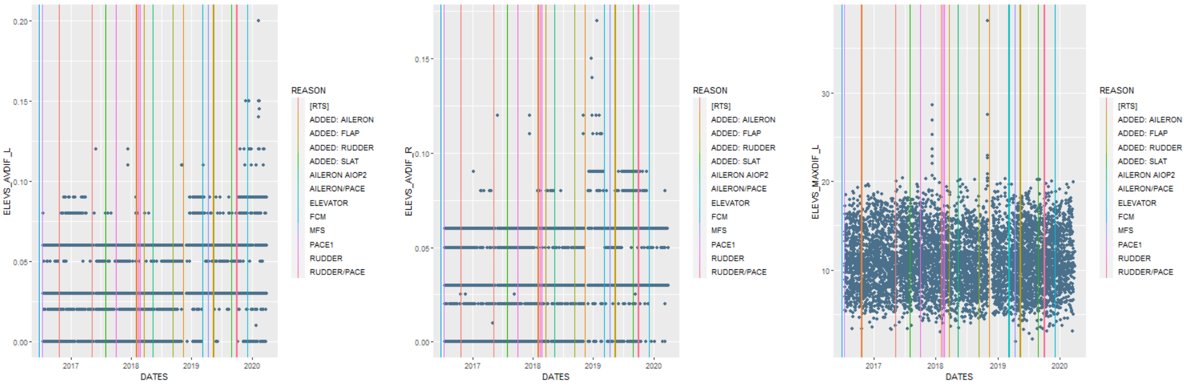
The ELEVS_AVDIF features have been identified as correlating to some faults in the dataset. The plots presented in Fig. 3.16a and Fig. 3.16b show an interesting example regarding this. Specifically, for the right-side elevator surface, Fig. 3.16b, it shows that the difference between readings of the inboard and outboard sensors is seldom greater than 0.1° in magnitude, and when it is, the same happens for the left-side elevator sensors. However, this is not the case for the period starting from the end of 2018, where it is visible for the right side an increase in sensor reading discrepancy, which only ends after a fault in the elevator subsystem, which was acted upon by replacing the right inboard elevator sensor.

Moreover, on the left side variable, an increase in discrepancy is also visible by the end of 2019, and it would not be illegitimate to think that there might be an impending fault concerning either of the left side sensors, however by this time the data is considerably more scarce due to there being less flights.

Given this, both ELEVS_AVDIF_L and ELEVS_AVDIF_R were kept in the dataset.

Regarding the ELEVS_MAXDIF_L and ELEVS_MAXDIF_R features (ELEVS_MAXDIF_L being represented in Fig. 3.16c for a specific aircraft), they are both random with unchanging behaviour across all aircraft safe for the system-wide peaks, which means they do not carry any additional information, so they were discarded.

The ELEVS_MINDIF features were both found to be a constant of zero in all aircraft, so they were also removed from the dataset.



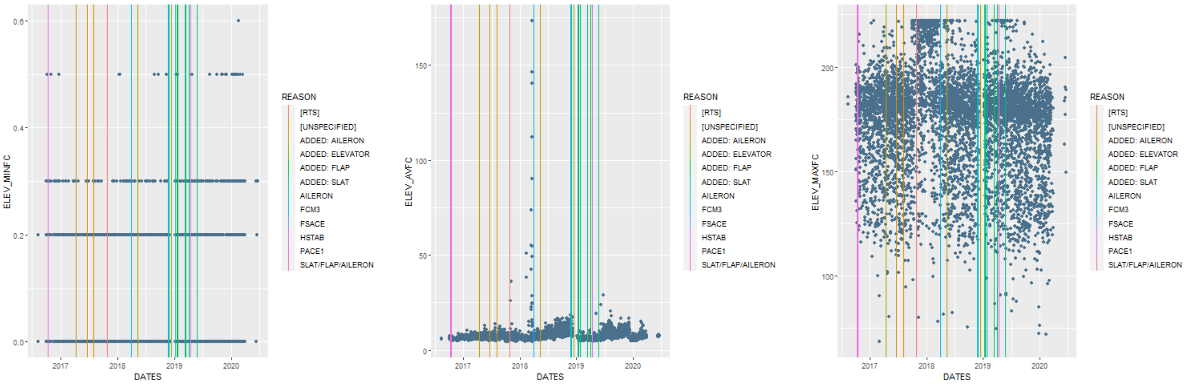
(a) (b) (c)

Figure 3.16: Plots for the ELEVS_AVDIF_L (left), ELEVS_AVDIF_R (centre), and ELEVS_MAXDIF_L (right) features on a single aircraft

Finally, regarding the elevator force features, both ELEV_MINFC (Fig. 3.17a) and ELEV_MAXFC (Fig. 3.17c) seem to carry no meaningful information regarding faults or system degradation, despite there being some behaviour changes. For instance, in Fig. 3.17c, there are clear differing distributions: one starting at the last quarter of 2017 and ending around the first quarter of 2018, and the other being the rest of the scatter plot — no reason was found for any of these behaviour changes in the maintenance reports (possibly the FCM3 fault marked the end of the abnormal behaviour, but it is not certain), and this happens in all other aircraft. In fact, ELEV_MAXFC is for the majority of aircraft random and these behaviour changes are very rare, yet when they exist, no source was found.

As for ELEV_AVFC, Fig. 3.17b shows not only the previously mentioned fault correlation to the FCM3, but also an increasing trend throughout 2018 ending in December of 2018 where some elevator components were found with physical damage, and were hence replaced (in the plot, this fault is marked as an Aileron fault, however there was also this one in the same day).

In conclusion, only ELEV_AVFC was kept, as it showed in some aircraft both degradation and correlation to faults on the Elevator and Electrical Systems.

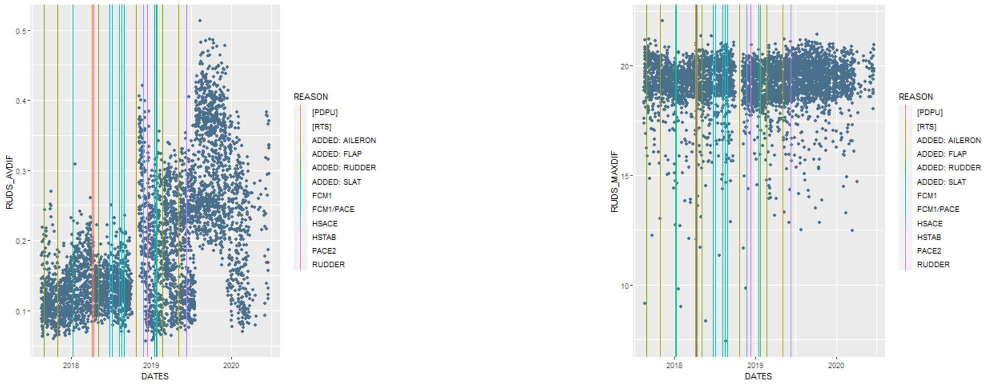


(a) (b) (c)

Figure 3.17: Plots for the ELEV_MINFC (left), ELEV_AVFC (centre), and ELEV_MAXFC (right) features on a single aircraft

Rudder

As for the rudder surface sensors reading disparity features (Fig. 3.18), RUDS_AVDIF was kept for its most evident apparent correlation to rudder faults. For instance, in Fig. 3.18a, an increasing trend from the last quarter of 2017 and extending towards the middle of 2018, seems to have led to, or at least was corrected after, successive faults on the rudder, followed by a rudder PCU replacement. Another example on the same plot would be the behaviour acquired after the maintenance check on the last quarter of 2018, comprised of relatively high values on the feature, closely followed by a fault of a P-ACE. What seems to compromise the usefulness of this feature, however, is the fact that not long after, it assumes values quite higher than those before the aforementioned faults, yet no other issues were reported. Still, it was kept. As for RUDS_MAXDIF, it seemed to bear no more information than its average counterpart; RUDS_MINDIF was always zero.

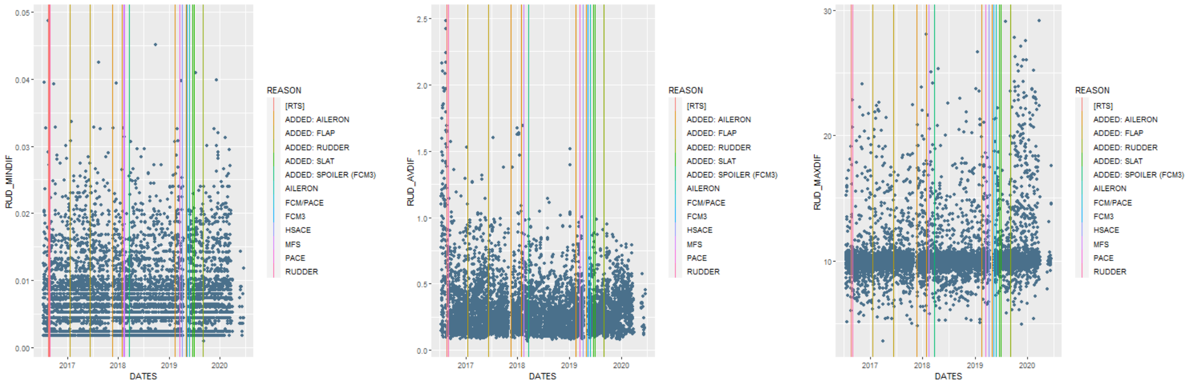


(a)

(b)

Figure 3.18: Plots for the RUDS_AVDIF (left), and RUDS_MAXDIF (right) features on a single aircraft

Regarding the rudder features accounting for its input/output relationship (see Fig. 3.19), only RUD_AVDIF was kept, as it displayed the greatest apparent correlation to rudder faults, the most flagrant example in the datset being represented in Fig. 3.19b, where the first faults in the studied period (rudder faults) seem to stem from the high values of this feature.



(a)

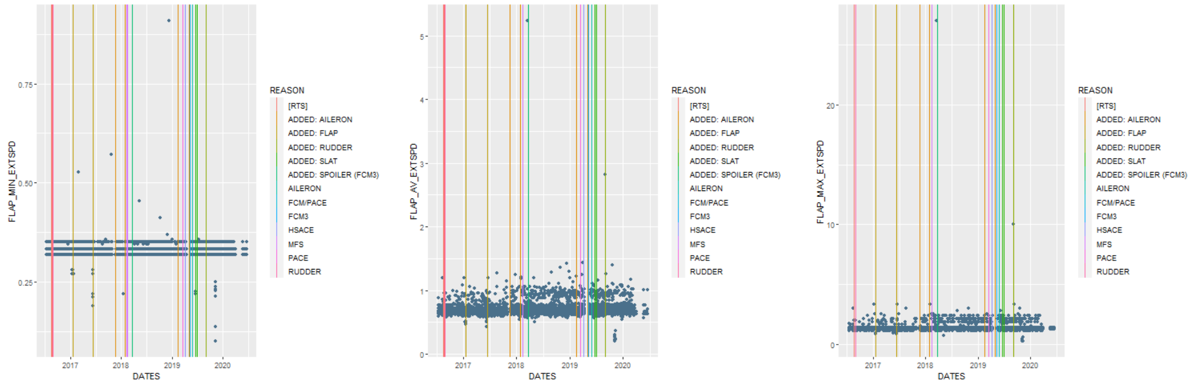
(b)

(c)

Figure 3.19: Plots for the RUD_MINDIF (left), RUD_AVDIF (centre), and RUD_MAXDIF (right) features on a single aircraft

Flaps/Slats

The flap and slat subsystems are here discussed together as their respective feature analyses are the same regarding fault correlation (flap feature plot characteristics leading to flap faults are the same as slat feature plot characteristics leading to slat faults, and in fact flap feature plots are generally quite similar to their slat equivalents).



(a) (b) (c)

Figure 3.20: Plots for the FLAP_MIN_EXTSPD (left), FLAP_AV_EXTSPD (centre), and FLAP_MAX_EXTSPD (right) features on a single aircraft

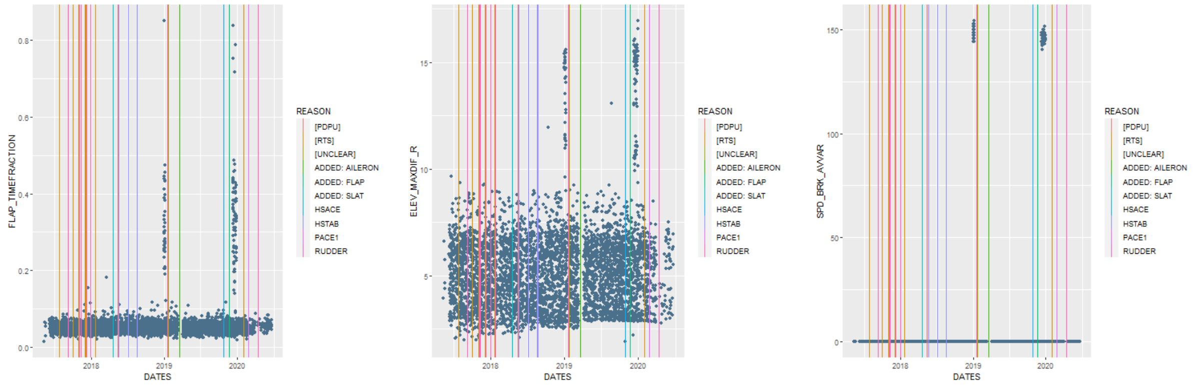
Of the features regarding the speed of the flaps to arrive at their correct deflection (Fig. 3.20), it was found that the retract and extend feature plots were very similar to each other in shape, differing only in scale. Of all of them, FLAP_MIN_EXTSPD was chosen to represent this system characteristic for two major reasons. On one side, for its clarity in fault correlation, visible in the two flap faults in 2017 in Fig. 3.20a with their respective outliers in the plot, indicating the speeds were found to be lower than normal, which is consistent with the fact that most of the system messages found on reports for these kinds of fault mentioned “surface jammed” or “underspeed”. On the other side, the number of possible normal values for this feature was remarkably lower than for the other two features (the apparent three horizontal lines that are formed by the plot points versus the more evident scatter on the other plots) — this would make this feature’s healthy state easier for the autoencoder to learn.

Analogously, the chosen feature for the slats was SLAT_MIN_EXTSPD.

As mentioned before, these speed features were found to be prone to missing values, and initially this was thought to be due to very short flights — such as rejected take-offs, — where the flaps and slats are barely used, if at all. However this was found not to be the case, which prompted individual inspection of the flights which were causing the issue. These flight raw data files, along with the discrete nature of the flap sensor readings, provided a glance at the nature of the aforementioned system-wide peaks.

Indeed, it was found that for these troublesome flights, the sensors were reading both flaps and slats to move irrespective of the flap/slat lever position: the lever could show to stay in position, yet the flap and slat surfaces would move, and it was not possible to calculate a speed of arrival at the theoretical deflection, because there was no means to compute that deflection without lever movement

— hence the missing value. So when circumventing this issue by creating the FLAP_TIMEFRACTION and SLAT_TIMEFRACTION features, the following was found, as shown in Fig. 3.21:



(a) (b) (c)

Figure 3.21: Plots for the FLAP_TIMEFRACTION (left), ELEV_MAXDIF_R (centre), and SPD_BRK_AVVAR (right) features on a single aircraft

The problematic flights belonged to the outliers which represent the system-wide peaks. This figure shows plots representing features for three distinct subsystems, all having this anomaly, and in truth it affected all subsystems.

This discovery helped shedding light on the credibility of these data points: if something as critical as the control surfaces moving on their own were happening, across multiple systems, no aircraft would be flying in any way, shape, or form. In fact, it would have been something the pilots would be aware of, from the very beginning of the flight — in the taxi-out phase, where the controls are tested and the surfaces are checked for movement.

Along with the fact that no reports show any such critical complaints around these periods, and in addition the fact that this seemed to happen at around the same period for all aircraft (see Fig. 3.22), this lead to the conclusion that the issue was not in any of the studied systems, and should instead be either a recording, or a software concern.

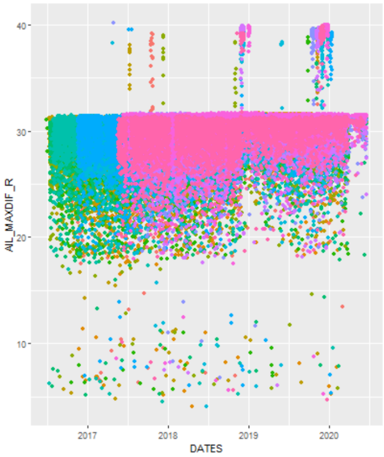


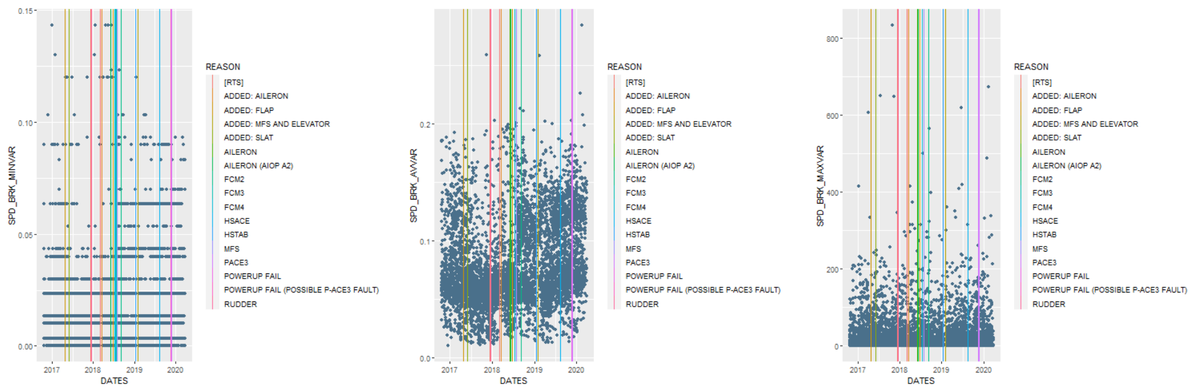
Figure 3.22: Plot for the AIL_MAXDIF_R feature for all aircraft, each color representing one aircraft; most system-wide peaks appear at around the same periods

Due to this, it was concluded that system-wide peaks are not representative of the system's health, and that faults happening seemingly connected to them was largely coincidental, as supported by the fact that the vast majority of peaks either is not preceding any fault, or is preceding a fault that is not remotely as critical as this event would be, were it a true fault (for instance, Fig. 3.21's peak at the beginning of 2019 seemingly relating to a rudder fault, yet it is known that this fault was exclusive to the rudder).

This is the reason features exhibiting this characteristic were avoided whenever possible, as it was understood the peaks were polluting the data. The same reasoning lead to the discarding of the FLAP_TIMEFRACTION and SLAT_TIMEFRACTION features, with the missing values on the flap and slat speed features being replaced with their respective medians.

Spoilers and Speed Brakes

The three SPD_BRK features are illustrated in Fig. 3.23, and of them SPD_BRK_MINVAR most evidently showed correlation to multifunction spoiler faults: as exemplified in Fig. 3.23a, the first half of 2018 displays a high frequency of higher values than usual, followed by several FCM faults, and finally a multifunction spoiler fault (marked "MFS" in the plot legend). Moreover, SPD_BRK_AVVAR might have also been a choice, however it displays in some aircraft a single outlier several orders of magnitude greater in value than the rest of the plot — likely connected to the system wide peaks — so it was not used.

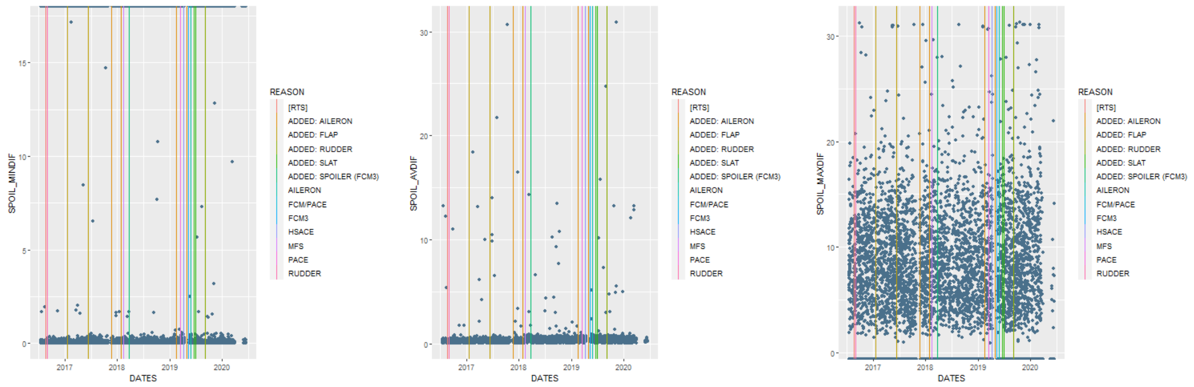


(a) (b) (c)

Figure 3.23: Plots for the SPD_BRK_MINVAR (left), SPD_BRK_AVVAR (centre), and SPD_BRK_MAXVAR (right) features on a single aircraft

SPOIL_AVDIF, SPOIL_MAXDIF and SPOIL_MINDIF (see Fig. 3.24) all share the same problem, which is that they rely on pilot input to be able to be calculated. The speed brake subsystem can be either controlled by the pilots, or the auto-pilot, and information about which of these is the agent on the position of the surfaces is given by the AGS variable SPD_BRK, which is 0 when the agent is the auto-pilot, and 1 when it is either of the pilots. The issue arises from the fact that it is not unusual that in a flight the SPD_BRK variable is always 0, meaning there was no instant where the pilots moved the speed brake lever. As such, for these cases, a difference between expected and actual surface deflections cannot be calculated, which then translates into missing values on these three features. The frequency of these can be seen at the top of the plot for SPOIL_MINDIF (Fig. 3.24a), and at the bottom of the plot for SPOIL_MAXDIF (Fig. 3.24c) for they turn into $+\infty$ and $-\infty$ for these features, respectively. As visible, these cases are quite numerous. In fact, almost 50% of all flights carry this characteristic.

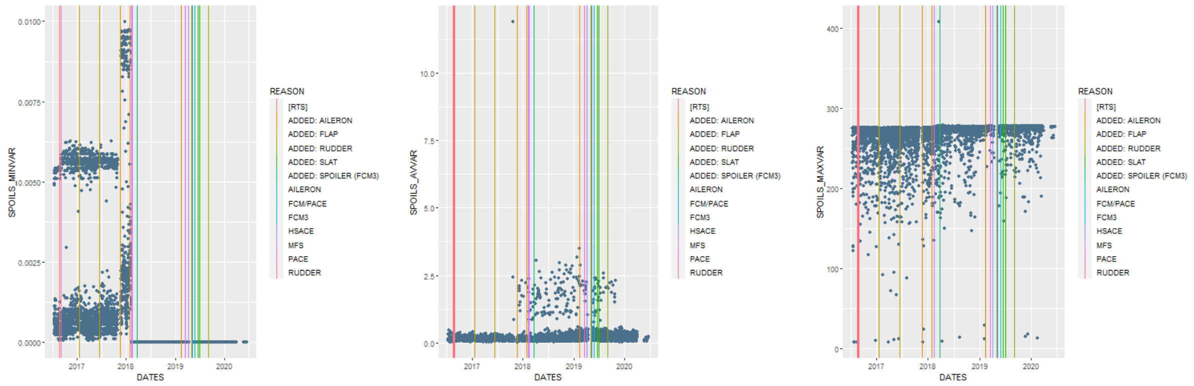
Given this, and the fact that no obvious visual correlation to spoiler faults was found in any of these features in any aircraft, they were all discarded.



(a) (b) (c)

Figure 3.24: Plots for the SPOIL_MINDIF (left), SPOIL_AVDIF (centre), and SPOIL_MAXDIF (right) features on a single aircraft

Of the spoiler sensor reading variance features (Fig. 3.25), SPOILS_MINVAR was the one quite notably most representative of fault imminence, for all aircraft. Fig. 3.25a shows quite an illustrative example on the matter, where a sudden increase in the values for this feature leads to multiple spoiler faults a few months later, the fault being corrected by the replacement of a PCU. For this reason, it was the only one kept.



(a) (b) (c)

Figure 3.25: Plots for the SPOILS_MINVAR (left), SPOILS_AVVAR (centre), and SPOILS_MAXVAR (right) features on a single aircraft

3.3.4 Test and Validation Set Creation

In order to test the algorithm, it was not enough to only have a set of data with which to train the model, that is to say the train set. Rather, it was also necessary to have data to test the model — test set —, and data with which to tune the hyper-parameters (such as λ, α, and t_{max} discussed in Section 3.1 - Algorithm Overview), called the validation set.

To do this, both the validation set, and the test set needed to be comprised of sequences which had not yet led to failure, and still the true RUL had to be known. This was accomplished such that the sequences corresponding to the test set were simply sequences taken from the original dataset, but truncated at random, so the algorithm never sees the discarded part, but the true RUL is known so as to gauge the model's performance. The same method was applied to the validation set.

3.3.5 Data Normalising

Having selected which features to keep, the dataset was still comprised of features of greatly different scales, which could cause problems during modelling [24]. In this case, an example of great disparity in scale would be comparing a maximum force feature, ranging from $\sim$100lbf to $\sim$400lbf, to a minimum variance feature, ranging from 0 to $\sim$0.01. With such variety in scale, normalisation of the data had to be done.

The method used to deal with this was the Min-Max normalisation, in order to linearly re-scale every feature into values in the interval $[0, 1]$, as shown in the following expression:

$$z = \frac{x - \text{min}(x)}{\text{max}(x) - \text{min}(x)}, \tag{3.6}$$

where z is the re-scaled feature, x is the original feature, and min and max are the minimum and maximum values found for feature x, respectively.

Additionally, the Z-score normalisation method was also tested:

$$z = \frac{x - \mu}{\sigma}, \tag{3.7}$$

where μ and σ are the mean and standard deviations of the original feature, so that the normalised features have a distribution with a mean of 0 and a standard deviation of 1. However, both the computing time for training the autoencoder and the reconstruction quality were found to be better with the Min-Max normalisation method.

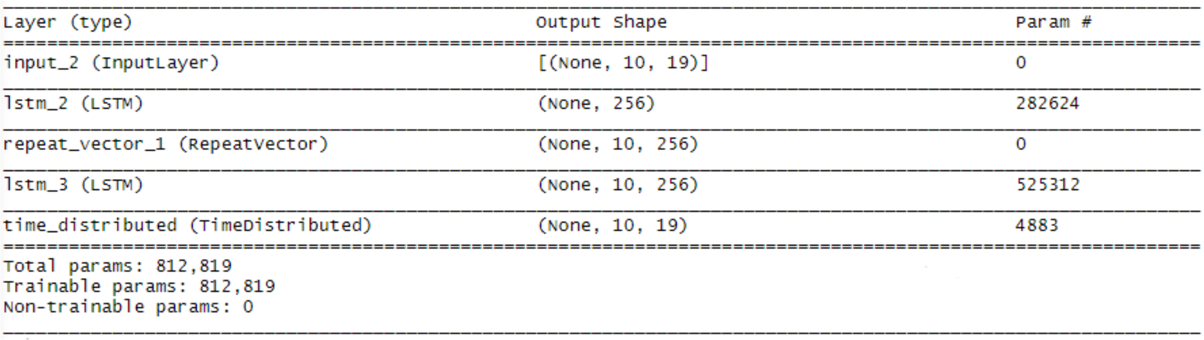
It should be noted that, for each feature, the statistics required for the normalisation were determined from the training data only, and then applied to all the data, so that both validation and test data were completely unseen by the model, and were also normalised to the same standard as the train data.

3.4 Autoencoder

The autoencoder is the key element of the algorithm. By training it to reconstruct data pertaining only to flights where the Flight Control System was assumed to be healthy, it is expected to display a greater reconstruction error when tasked with reconstructing cycles where this system is degraded.

The used autoencoder was built in the R programming language, making use of the Keras machine learning library. The encoder consisted of one LSTM layer with the Rectified Linear activation function and with a vector of 256 units as the output — 256 being the found number of units to be the best balance between computing time and reconstruction results. The decoder consisted of an LSTM layer with the ReLU activation function and with a 2D array with shape (*number of timesteps*, 256) as its output, followed by a time distributed layer so that the output of the autoencoder had the same shape as the original input. Moreover, between the encoder and decoder a repeat vector layer is required, so that the input of the decoder has the required shape for the decoder's LSTM layer (a 2D array rather than a vector).

The autoencoder model is hence as described in Fig. 3.26, considering a number of timesteps of 10, and a total number of features of 19.



```
________________________________________________________________________________________________
Layer (type)                          Output Shape                        Param #
================================================================================================
input_2 (InputLayer)                  [(None, 10, 19)]                    0
________________________________________________________________________________________________
lstm_2 (LSTM)                         (None, 256)                         282624
________________________________________________________________________________________________
repeat_vector_1 (RepeatVector)        (None, 10, 256)                     0
________________________________________________________________________________________________
lstm_3 (LSTM)                         (None, 10, 256)                     525312
________________________________________________________________________________________________
time_distributed (TimeDistributed)    (None, 10, 19)                      4883
================================================================================================
Total params: 812,819
Trainable params: 812,819
Non-trainable params: 0
________________________________________________________________________________________________
```

Figure 3.26: Layers of the Autoencoder, as summarized by R

The LSTM layers expect a 2D array per sample as the input, so the input comprised of a batch of samples must be a 3D array. For the first layer of the autoencoder, this array — the input data — had dimensions of (*batch size*, *number of timesteps*, *number of features*). This meant that the run-to-failure sequences had to be organised in an array in the form (sample, time cycle, feature), which is to say in this case (sequence, flight, feature).

Not all sequences in the dataset were used to train the autoencoder, as the dataset was divided into 3 groups — train, validation, and test sets — with ratios of 70% – 15% – 15% of the total data, respectively. During training, the autoencoder would only see the train set, and in fact, only a number of initial flights per sequence, as these were the ones assumed healthy. The choice for this number took into account a few major factors:

- It would be important to have a high number of flights with which to train so that the model would better learn the peculiarities in some of the features, however the number could not be too high otherwise the training might become too influenced by degraded flights;

- This number sets the dimensions of the input of the autoencoder, so it also sets the minimum allowed length of a run-to-failure sequence — smaller lengths would not be accepted by the model — so setting too high a value may deem too many sequences unusable;
- Finally, and for the same reason as the last point, in a real life application, when wanting to learn of the health of an aircraft, this number also dictates the minimum allowable number of flights used to gauge that health — so too high a value means personnel have to wait until the aircraft of interest has fulfilled those many flights.

With all this being set, the autoencoder model was then compiled and trained with the Mean Squared Error as the loss function, and the Adam optimiser, on 1000 epochs.

It was then tasked with reconstructing the sequences of the train, validation, and test sets. When reconstructing the original data, the shape of the input to the autoencoder must be the same as the same as the shape of the data it was trained on, e.g. if it was trained on sequences of 10 flights, it is necessary to reconstruct each sequence of the total data every 10 flights and gather the total results. If this were to be done by simply reconstructing the first 10 flights, then the next 10 and so on, the results would be disjointed and have no continuity, because the algorithm does not know that the second set of 10 is related to the first set of 10.

This is worked around by introducing a sliding window: the autoencoder reconstructs the first set of 10, then instead of jumping 10 flights to the next set, it jumps a smaller amount so that there are common flights in every set. The results for flights that have been reconstructed more than once are then averaged out.

3.5 Data Post-Processing

After obtaining the reconstructions of the train, validation, and test sets, the reconstruction errors throughout each sequence can then be computed. Considering a time series $Z = [z_1, z_2, ..., z_t, ..., z_L]$ for a sequence u with length L (where z_t is the vector of the features at time t), and considering its reconstruction $Z' = [z'_1, z'_2, ..., z'_t, ..., z'_L]$, this error $e_t^{(u)}$ is given by:

$$e_t^{(u)} = ||z_t^{(u)} - z_t'^{(u)}||. \tag{3.8}$$

Alternatively, a squared error can be considered so that larger reconstruction error later results in a much smaller health index:

$$e_t^{(u)} = ||z_t^{(u)} - z_t'^{(u)}||^2, \tag{3.9}$$

and both versions were tested, the best being chosen via a validation set.

Either of these error measurements would be normalised so as to obtain a health index $h_t^{(u)}$, as mentioned before:

$$h_t^{(u)} = \frac{e_M^{(u)} - e_t^{(u)}}{e_M^{(u)} - e_m^{(u)}}, \tag{3.1}$$

where $e_M^{(u)}$ and $e_m^{(u)}$ correspond to the maximum and minimum occurring errors in the reconstruction of sequence u.

From calculating this health index throughout the length of the sequence, for all sequences, resulted the degradation curves $H^{(u)}$. Moreover, further processing of these curves would allow for the creation of three possible scenarios for the curve matching phase:

1. The curves were used as-is with no further processing;

2. A moving average was performed on the curves as they were originally very noisy, and

3. A linear regression with the original curves as the target, and the original features as the estimators was used in order to create more consistent curves for the matching phase.

Once again, here the best results of a validation set dictated which of these scenarios to use on the test set.

Finally, the similarity between test curves and train curves was computed and an estimation of the RUL for each test sequence was made, as described in Section 3.1 - Algorithm Overview.

Chapter 4

Results

Following the methodology presented in the preceding chapter, the model was tested and various characteristics of it were still tuned to allow for better results. To summarise the flow of the algorithm in order to have an understanding of its adjustable aspects, the following list of steps that the model follows should be considered:

1. Gather a dataset from the available data (the run-to-failure sequences of flight summaries);

2. Divide the total data into train, validation, and test sets;

3. Define the Autoencoder, and train it with only the few initial flights from each sequence of the train set;

4. Reconstruct the full sequences of the train, validation, and test sets;

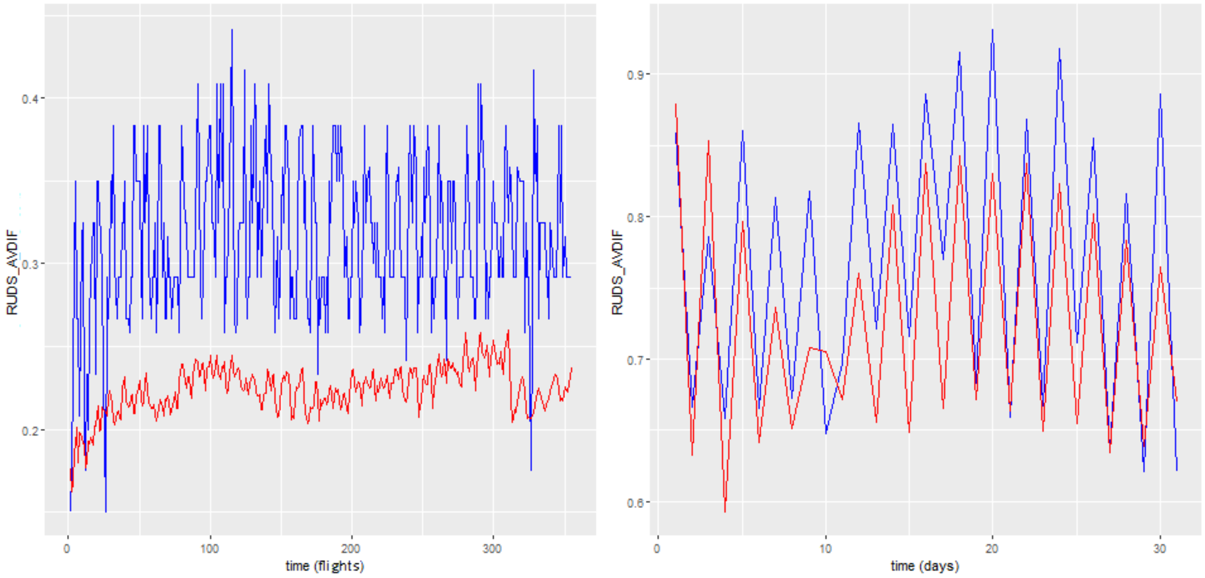
5. Compute the reconstruction error of these sequences and normalise it, to turn it into a degradation curve for each sequence;

6. Compare each curve from the validation set, to each curve from the train set, while varying the several parameters which needed to be configured, and estimate the RUL for each configuration, then choose the configuration with the best results;

7. Apply the obtained configuration to the test set and obtain the final RUL estimations.

The first obstacle in the procurement of adequate results manifested itself in Step 4 of this list, as the reconstruction of the various features of the sequences was found to be unsatisfying. This was attributed to how noisy the features seemed to be which made it more difficult for the autoencoder to find patterns in their behaviours, thus decreasing performance. The consequence of this problem was that the reconstruction error would not only stem from abnormalities in the data due to system degradation, but also simply from model inadequacy, thus deeming the results unreliable.

In an attempt to mitigate this issue, it was found that rearranging the data so that each cycle corresponded to a day rather than a flight (by averaging all flights for each day) was beneficial in the sense that it made each feature more predictable, hence easier to learn and reconstruct. This also allowed for lesser computing time, as now each sequence was both easier to learn, and shorter, hence faster to reconstruct. By virtue of this, features which the autoencoder was having difficulties reconstructing were now being reconstructed to near perfection whenever their behaviour corresponded to a non anomalous one. Fig. 4.1 shows an example of a feature benefiting from this change.



(a)

(b)

Figure 4.1: Examples of the reconstruction of the RUDS_AVIDIF feature in a sequence with the time cycle set as a single flight (left) and set as a day (right)

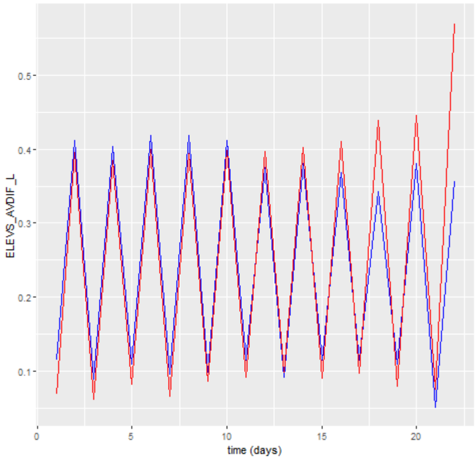


Figure 4.2: Example of near perfect reconstruction of one of the features which highly depend on which P-ACEs are engaged in each day

This means that henceforth all mentions to time cycles, including the RUL estimates, are no longer in flights, rather in days. As an addendum on this matter, it is of worthy note that the features to which this change was most favourable were also the features whose behaviours depend on which P-ACEs are used in a flight, due to these being switched mostly on a per-day basis. As such, these features now showed a much more predictable behaviour, hence no longer was there need to keep the ELACE_ENG variable, as the autoencoder was reconstructing these features perfectly adequately, as shown in Fig. 4.2.

The results may now be presented, and due to further difficulties in obtaining satisfactory RUL estimates, two sets of results were obtained (referred to as Base Results, and System-Wide-Peak-Filtered Results), as will show the following sections.

4.1 Base Results

As discussed before, prior to obtaining results on the test set, the validation set must be used to tune the parameters which are possible to vary, all of which pertaining to the stage where degradation curves are compared to each other. For convenience, these parameters are summarised here:

- Section 3.1 - Algorithm Overview introduced the parameters λ, α, and t_{max}, concerning the scale of the similarity between curves, the minimum similarity taken into account for the weighed average on the RUL estimation, and the maximum considered time-lag between two curves, respectively;
- Section 3.5 - Data Post-Processing introduced the reconstruction error which is used as a measure of health at time t, which can be taken as either the norm of the difference between the vectors of features and feature reconstruction (Eq. (3.8)), or it can be taken as the square of said norm (Eq. (3.9));
- Finally, Section 3.5 - Data Post-Processing also introduced the different ways in which the degradation curves can be further processed given the fact that they are very noisy: they can be used as-is (with no further processing), a Moving Average can be done on them, or the curves to use can be instead the result of a linear regression with the target curves as the original ones.

This meant that the algorithm was run with the validation set for every possible configuration with λ, α and t_{max} going through sets of values within a certain range. Each configuration was then evaluated and chosen based on the Mean Absolute Error (MAE), Root Mean Squared Error (RMSE) performance metrics as:

$$\mathit{MAE} = \frac{1}{n}\sum_{u=1}^{n}|\hat{R}^{(u)} - R^{(u)}| \tag{4.1}$$

and

$$\mathit{RMSE} = \sqrt{\frac{1}{n}\sum_{u=1}^{n}(\hat{R}^{(u)} - R^{(u)})^2}, \tag{4.2}$$

where n is the number of sequences in the set, and $\hat{R}^{(u)}$ and $R^{(u)}$ are the estimate and actual RULs for sequence u, respectively.

The number of False Positives and False Negatives in the estimates were also taken into account. A False Positive in this context was defined as the model underestimating the RUL by under 15 days, while a False Negative meant that the model overestimated the RUL by over 15 days. These two metrics were used to choose between configurations which might have similar MAE and RMSE, and moreover, preference was given to configurations which showed the least possible amount of False Positives, and only if this still was not enough to differentiate, the False Negatives were used. This was because it was understood that if the model must fail, then it was preferable for this to be a False Negative rather than a False Positive, given that in a real life application, a low RUL prediction should always be an indicative of degradation in the system, so as to not alarm the maintenance personnel of an issue which does not exist.

Given this, the best configuration found for the validation set was with the Degradation Curves smoothed with a Moving Average, $e_t^{(u)} = ||z_t^{(u)} - z_t'^{(u)}||^2$, $\lambda = 0.0005$, $\alpha = 0.74$, and $t_{max} = 50$. This corresponded to the estimates for the validation set presented in Fig. 4.3:

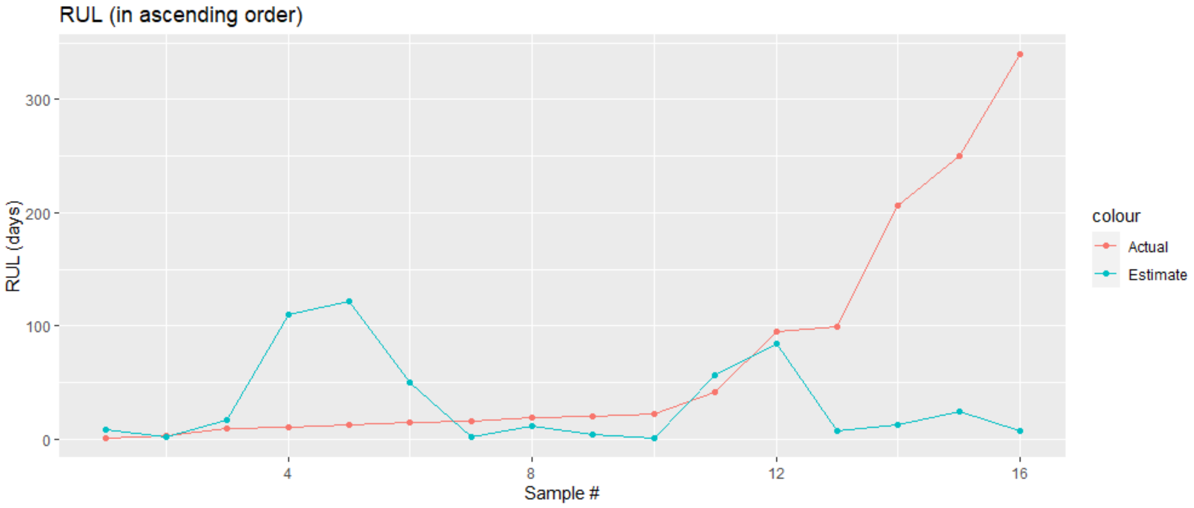


Figure 4.3: Actual and estimated RUL for the validation set, in ascending order of the actual RUL

And in turn, when the model was run with the chosen parameters on the test set, this corresponded to the results presented in Fig. 4.4:

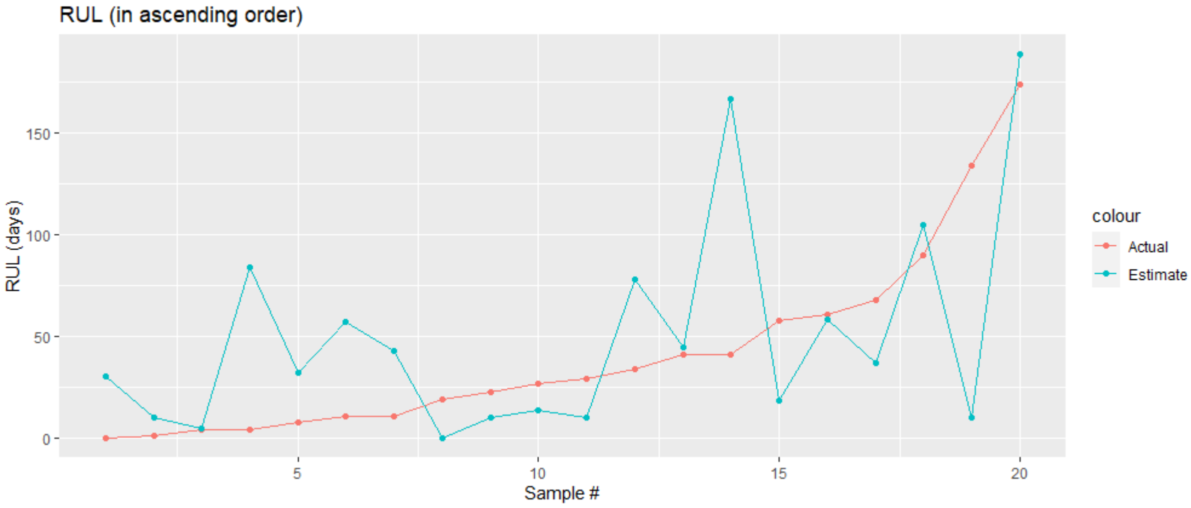


Figure 4.4: Actual and estimated RUL for the test set, in ascending order of the actual RUL

4.2 System-Wide-Peak-Filtered Results

As discussed before, the phenomenon here referred to as system-wide peaks comes in detriment to the data, and features were selected so it had the least possible impact to the dataset, and hence to the results. As will be discussed in Chapter 5 - Result Discussion, this was still not enough, so further action was taken against these occurrences.

Flights corresponding to these peaks were therefore removed from the dataset via a cutoff on the FLAP_TIMEFRACTION feature (which had been discarded from the dataset in the feature selection phase): flights which had too high a value for this feature were considered to be part of the outliers which form the peaks, and the cutoff was set to 0.15.

It should be noted that the reason this had not been done before, and furthermore the reason for which it is not a perfect solution, was that of the thirteen aircraft comprising the fleet, most displaying these peaks at some point or another of their useful life, only in six did the peak phenomenon display itself in the Flap or Slap subsystems, where either the FLAP_TIMEFRACTION or SLAT_TIMEFRACTION variables could be good candidates for setting a clear cutoff. For the rest of the fleet, these Flap and Slat features display good behaviours, and no good cutoff was found for any other feature, as these varied significantly not only across aircraft, but also within a single aircraft's lifecycle, so removing these flights would have led to removing also other flights which were not part of the problem.

With these matters present, the best configuration for the validation set was found to be with the Degradation Curves smoothed with a Moving Average, $e_t^{(u)} = ||z_t^{(u)} - z_t'^{(u)}||^2$, $\lambda = 0.0005$, $\alpha = 0.95$, and $t_{max} = 50$. This corresponded to the estimates for the validation set presented in Fig. 4.5:

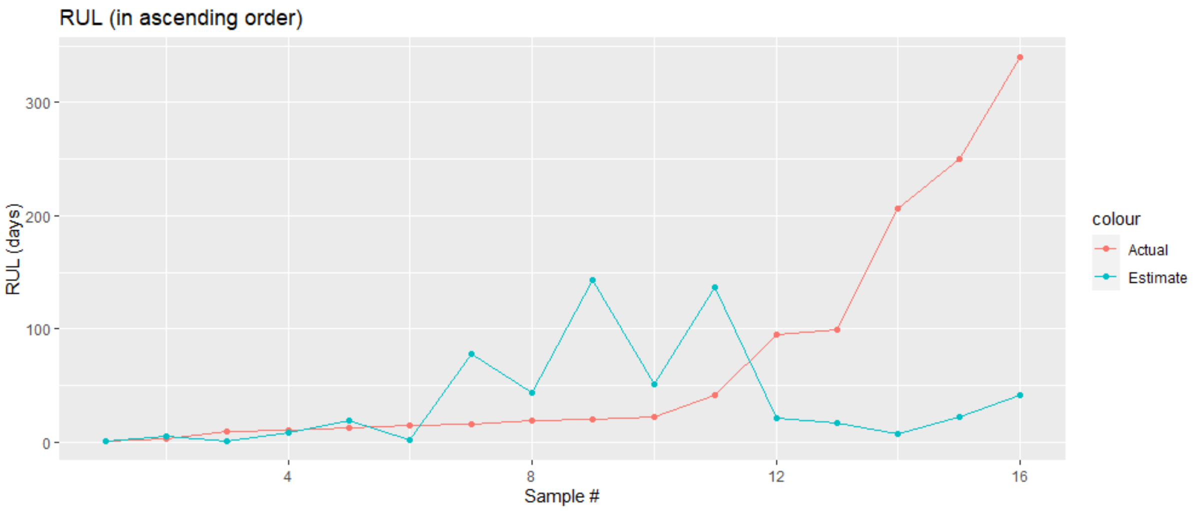


Figure 4.5: Actual and estimated RUL for the validation set, in ascending order of the actual RUL

And in turn, this configuration yielded the following results corresponding to the test set, presented in Fig. 4.6:

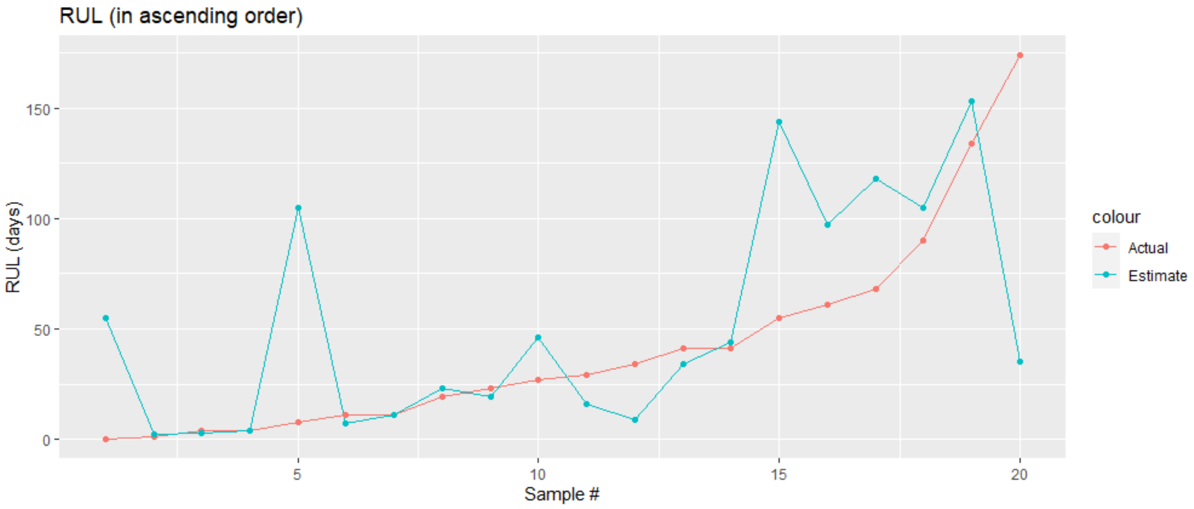


Figure 4.6: Actual and estimated RUL for the test set, in ascending order of the actual RUL

Chapter 5

Result Discussion

Having obtained the predictions of the Remaining Useful Life of the test set, the interest lies in understanding the reasons for which the model outputted a certain RUL estimate and not another. When the prediction was accurate, the algorithm should understand why the system failed, thence showing that in the features corresponding to the failed subsystem; in the case of a False Positive, there should be some basis for the algorithm to predict a shorter RUL than its actual value; and finally, for a False Negative, it interests to know why the model did not see any anomalies and thus overestimated the RUL of a sample sequence.

This is the analysis made in the following sections, where the effort is allocated towards discussing each of these three points, for both the results from Section 4.1 - Base Results, and from Section 4.2 - System-Wide-Peak-Filtered Results.

After this, a conclusion on the validity and generalisability of the results is presented.

5.1 Base Result Validation/Analysis

The Base Results, as shown in Section 4.1 - Base Results, and whose accuracy may be gauged via Table 5.2 and Table 5.1, showcased a model which was neither very accurate on the validation set, and even less so on the test set, so it is interesting to assess whether or not there is validity in any of these predictions.

The performance metrics ensuing the predictions follow in Table 5.1:

Table 5.1: Performance metrics on the RUL estimations from the base results

MAE	25.39
RMSE	31.19
Number of False Positives	5
Number of False Negatives	7

and Table 5.2 shows the obtained RUL estimations:

Table 5.2: Results of the base approach (RUL estimates rounded to the unit)

Sequence Number (from Fig. 4.4)	RUL Estimate (days)	Actual RUL (days)
1	31	0
2	10	1
3	5	4
4	84	4
5	32	8
6	57	11
7	43	11
8	0	19
9	10	23
10	14	27
11	10	29
12	78	34
13	45	41
14	167	41
15	18	58
16	58	61
17	37	68
18	105	90
19	10	134
20	189	174

5.1.1 True Positive Example

The chosen sequence to represent the True Positive case was sequence number 16, for being the most accurate on a high actual RUL value. The sequence corresponded to a fault on MAU1, and was originally 198 days long, and after being truncated, the model was only shown the first 137 days. The Degradation Curve computed by the model corresponding to this run-to-failure sequence is presented in Fig. 5.1, which shows somewhat of a decreasing trend, and is also marked by two sudden drops in health. The first drop can be related to the features pertaining to the number of important messages in the Electrical, Elevator, and Rudder subsystems (i.e. high values on the ELECSYS_MSG, ELEV_MSG, and RUD_MSG features, respectively, of which it is of worthy note that the value on the ELECSYS_MSG feature corresponded to this feature's highest achieved value in the entire dataset). On the other hand, the second drastic health drop corresponded to high values on the ELEV_MSG and RUD_MSG features in that day.

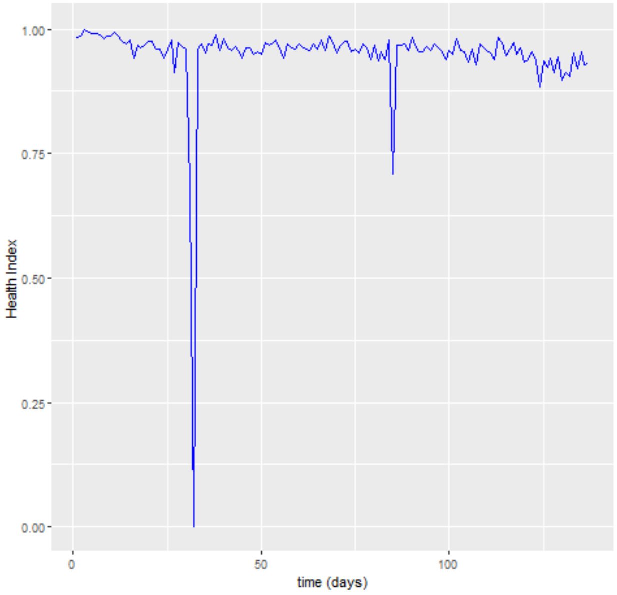
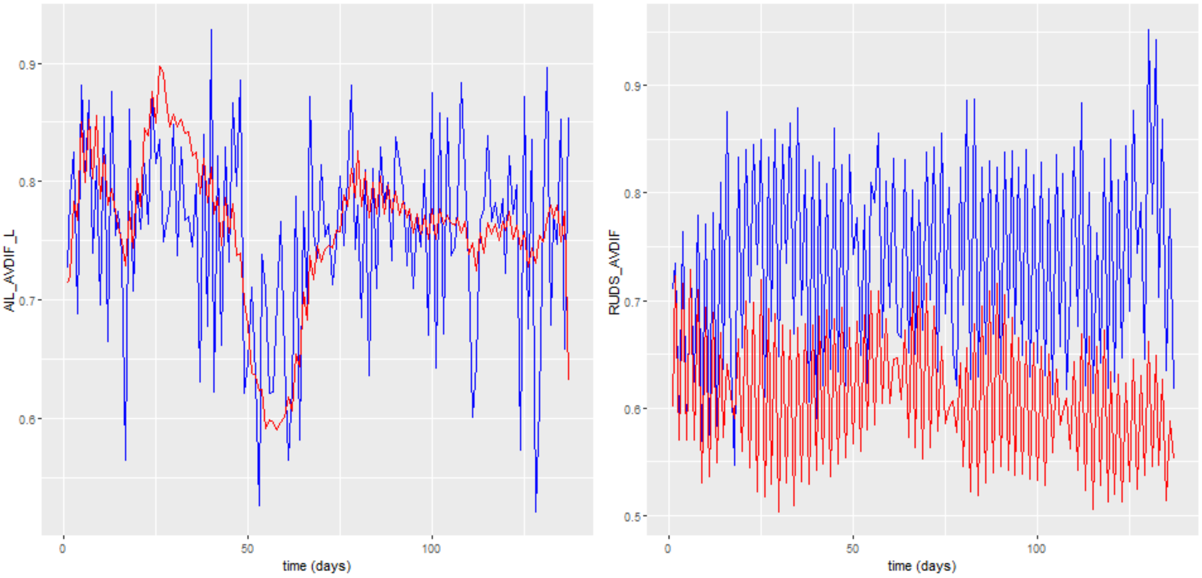


Figure 5.1: Degradation Curve for the sequence in analysis

In addition to these characteristics, inspection of the reconstructions of the sequence's features show that the autoencoder reasonably reconstructed features independent from the Electrical subsystem, such as aileron features (see Fig. 5.2a), yet had difficulties following the evolution of features highly dependent on the Electrical subsystem, such as rudder features (see Fig. 5.2b).



(a)

(b)

Figure 5.2: AIL_AVDIF_L (left) and RUDS_AVDIF (right) features, in blue, and their respective reconstructions, in red

This seems to indicate the model recognised the anomaly and estimated based on it.

5.1.2 False Positive Example

The most prominent case of a False Positive in Table 5.2 pertains to sequence number 19, which was originally 158 days long, while the model was only shown the first 24 days. The fault the sequence corresponds to pertains to FCM1. The corresponding Degradation Curve is shown in Fig. 5.3.

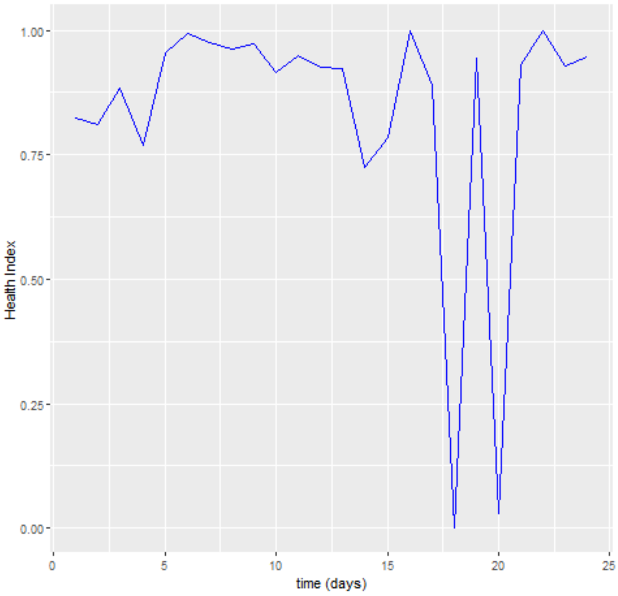
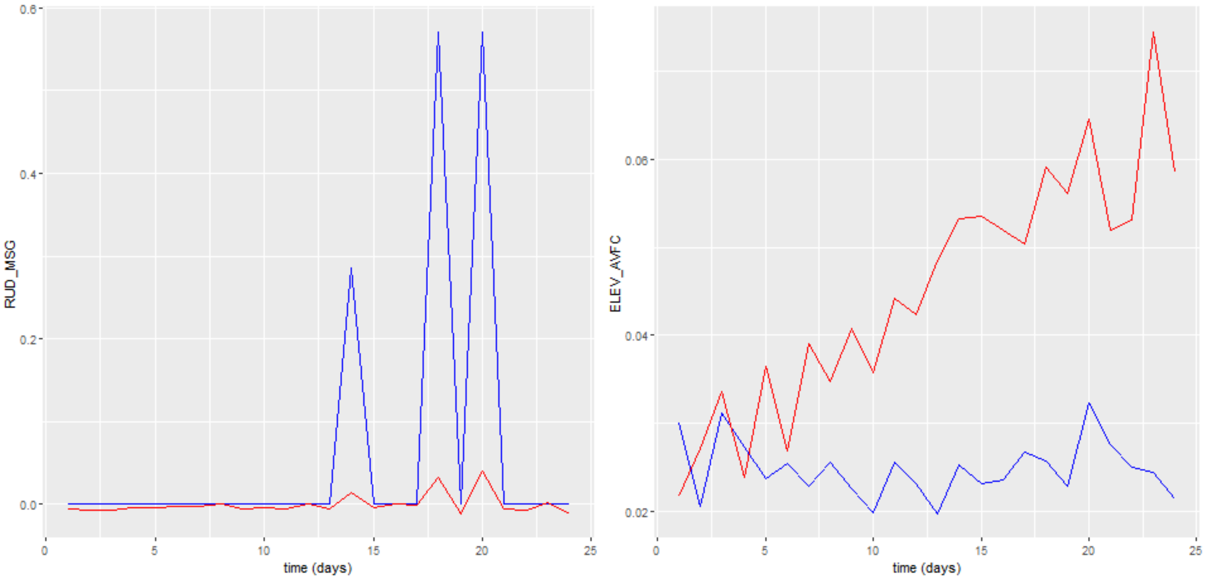


Figure 5.3: Degradation Curve for the sequence in analysis

The two visible sudden drops in health were concluded to stem from high values on the ELEV_MSG and RUD_MSG features (see Fig. 5.4a).



(a)

(b)

Figure 5.4: RUD_MSG (left) and ELEV_AVFC (right) features, in blue, and their respective reconstructions, in red

Moreover, the only feature showing a particularly poor reconstruction was ELEV_AVFC, indicating that the model identified its behaviour as an abnormal one. However, when inspecting maintenance reports in the period which encompasses the sequence and around the time the model estimated there would be a fault, no justification was found for this performance of the algorithm. There were two accounts of a current peak causing many Electrical systems in the aircraft to generate many CMC messages, yet no complaints or reports on the Elevator subsystem.

It seems, then, to be concludable that in this case the model failed to identify anomalies in the system, and its RUL estimate was both inaccurate, and with no connection to the real operation of the specific aircraft.

5.1.3 False Negative Example

As representative of the False Negatives in this set of results, sequence number 4 from Table 5.2 was chosen, corresponding to a fault on the flaps, originally 17 days long, the model being only provided with the first 13 days.

The respective Degradation Curve is shown in Fig. 5.5, whose health drops correspond to a high count of rudder related CMC messages (see Fig. 5.6a). Besides these drops, no degradation trend is visible in this curve.

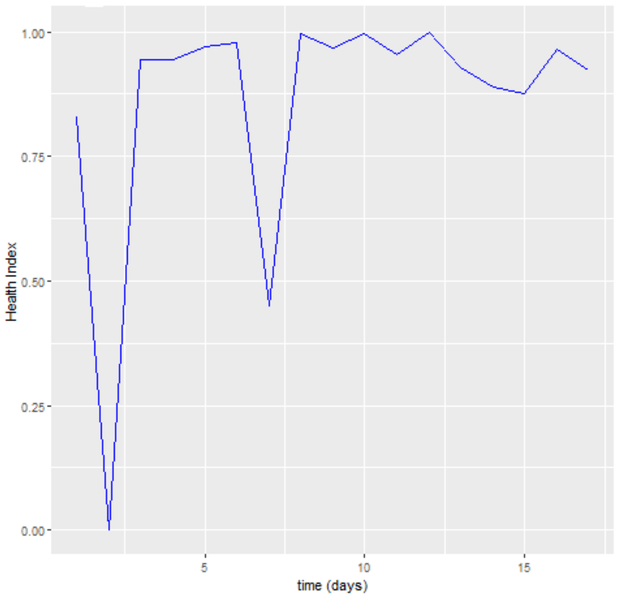
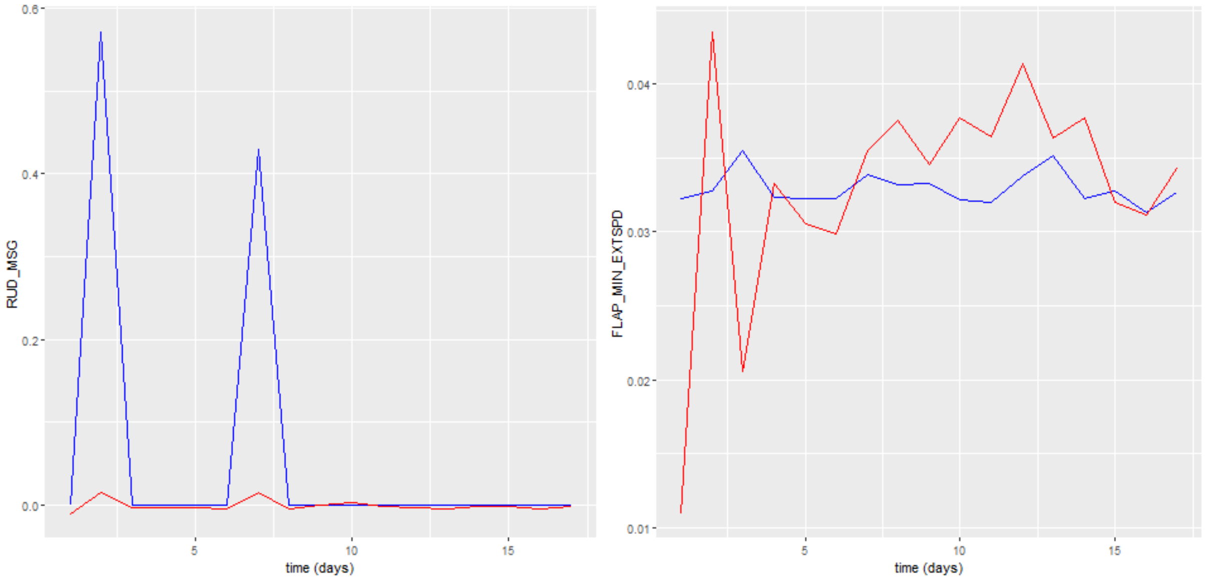


Figure 5.5: Degradation Curve for the sequence in analysis

Furthermore, when inspecting the Flap feature and its respective reconstruction (Fig. 5.6b), it is visible that the reconstruction is reasonable, and does not display an acknowledgement from the model that there is a flap anomaly. Reconstructions of other features were equally without trouble.



(a)

(b)

Figure 5.6: RUD_MSG (left) and FLAP_AV_EXTSPD (right) features, in blue, and their respective reconstructions, in red

In this case, the inaccurate RUL estimation seems to then stem from inadequate or insufficiently descriptive features for the flaps, as the model was not able to associate the existing features' behaviours to anomalies, and moreover, the low values present in FLAP_AV_EXTSPD's normalised plot in Fig. 5.6b suggest that this behaviour was indeed not anomalous, so this feature failed to translate the degradation of the flap subsystem, in this case.

It is also possible that degradation only began much closer to the day of the fault, in which case it would simply not be possible for the model to accurately predict the RUL, with the length of sequence it was given.

5.2 System-Wide-Peak-Filtered Result Validation/Analysis

The system-wide-peak-filtered results show a substantial improvement to the base results, confirming the assertion that the system-wide peaks were not consequential to system degradation (at least not Flight Control System degradation, yet perhaps some other system not in study in this work).

The performance metrics ensuing from these predictions follow in Table 5.3:

Table 5.3: Performance metrics on the RUL estimations from the system-wide-peak-filtered results

MAE	20.04
RMSE	27.97
Number of False Positives	2
Number of False Negatives	7

and the results are presented in the Table 5.4, for reading convenience:

Table 5.4: Results of the system-wide-peak-filtered approach (RUL estimates rounded to the unit)

Sequence Number (from Fig. 4.6)	**RUL Estimate (days)**	**Actual RUL (days)**
1	55	0
2	2	1
3	3	4
4	4	4
5	105	8
6	7	11
7	11	11
8	22	19
9	19	23
10	46	27
11	16	29
12	9	34
13	34	41
14	44	41
15	144	55
16	97	61
17	118	68
18	105	90
19	153	134
20	35	174

With the exception of sequences number 1 and 5 from Table 5.4, it is visible that the model was quite accurate predicting RULs up to the 9th entry, that is to say, up to an actual RUL of 23 days. From there, a decrease in general accuracy is visible, showing it is easier for the model to predict failures the closest they are to happening.

5.2.1 True Positive Example

The sequence consisting of a True Positive prediction which seems to raise most interest was sequence number 14 from Table 5.4, for being the most accurate outside of the first nine entries of that table. The sequence at hand corresponds to a fault in one of the rudder pedals and it is 119 days long, while the model was only provided with the first 78 days worth of data.

Fig. 5.7 represents the corresponding degradation curve which the model computed.

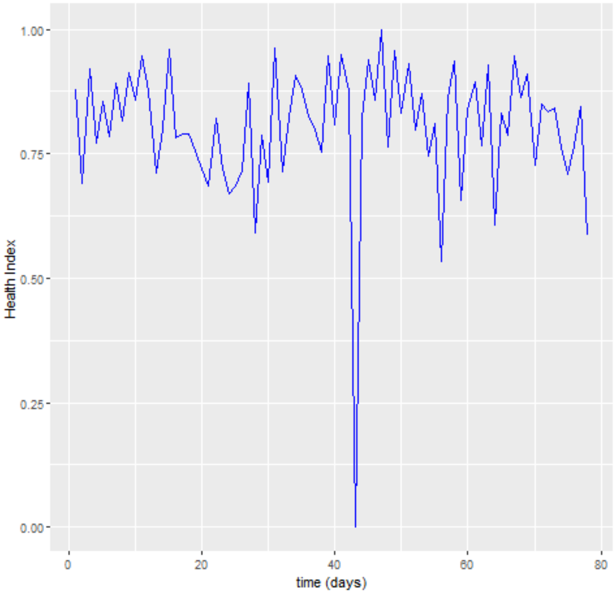
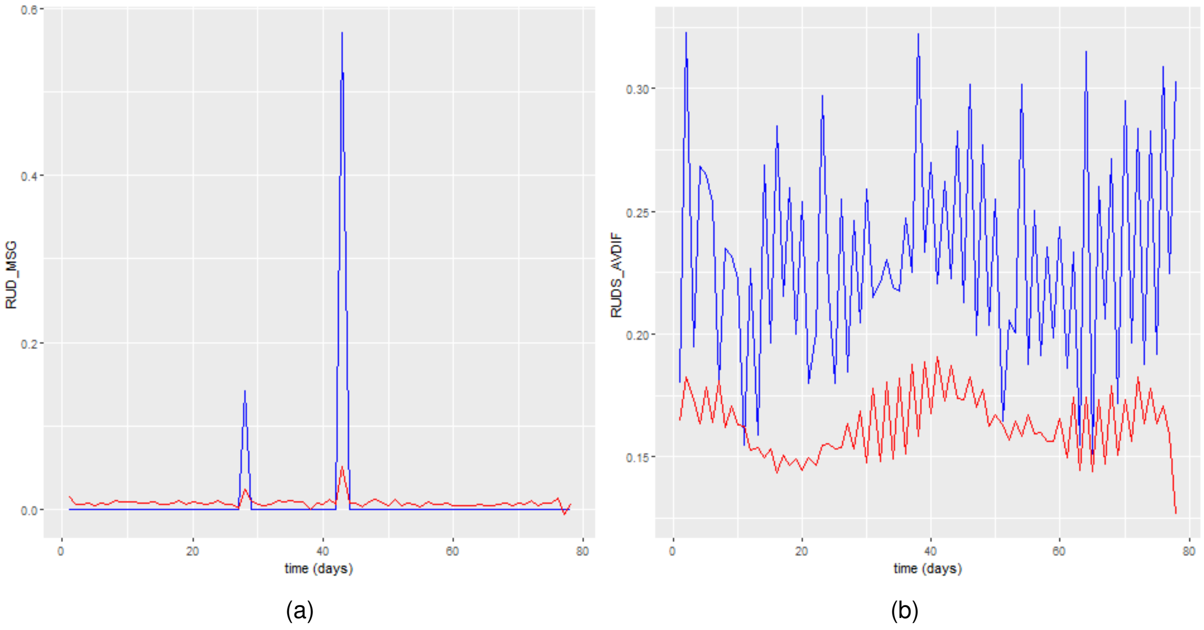


Figure 5.7: Degradation Curve for the sequence in analysis

Despite there not being a clear degradation trend, rather only a very slight one if any, the most prominent characteristic of that curve is the sudden drop in health at the 43rd day, which then goes back up. After inspection of the reconstructed features, it was concluded that this drop stems from an appearance of a high count of important rudder related messages on that same day (which is to say a high value on the RUD_MSG feature), as visible in Fig. 5.8a. This high count was also visible in the feature pertaining to messages from the Electrical System, although not quite as dramatically.



(a)

(b)

Figure 5.8: RUD_MSG (left) and RUDS_AVDIF (right) features, in blue, and their respective reconstructions, in red

Moreover, despite generally good reconstructions in most features in the sequence, the RUDS_AVDIF feature's reconstruction (see Fig. 5.8b) was remarkably always lower in value than its original plot, seemingly indicating that the model took issue with it, indicating a possibly anomalous behaviour. This is in spite of the original features values not being that high, although in inspecting other sequences, it is visible that the normalised plots for RUDS_AVDIF rarely go above the value 0.3. While this is not enough to justify an underestimation of the values in the plot at hand (which were mostly below 0.3 as well), it should also be kept in mind that the autoencoder does not simply learn the patterns of each feature, rather also relationships between features. It is possible that, given the plots for the other features, it expected lower values in this one.

5.2.2 False Positive Example

Sequence number 20 from Table 5.4 shows the most evident example of the model underestimating the RUL in the test set. The sequence in question was originally 203 days long, yet it was truncated so that the model only saw the first 29 days. The fault it pertains to took place in the ailerons.

The simplest explanation for the prediction to be so far from reality here would evidently be that an actual RUL of 174 days is too large and many a thing could happen to the system between the last point in the sequence the model has access to and the day of the fault, which could change the value of the prediction. However, and granted what has been stated before regarding preference towards False Negatives as opposed to False Positives, it cannot be ignored that the algorithm identified what it interpreted as an abnormal behaviour in the sequence's features, otherwise the predicted RUL might have not been so low a value.

For this reason, it is of interest to look for such abnormal behaviours and see if they might have had any impact in the operation of the corresponding aircraft.

The degradation curve computed by the model is presented in Fig. 5.9.

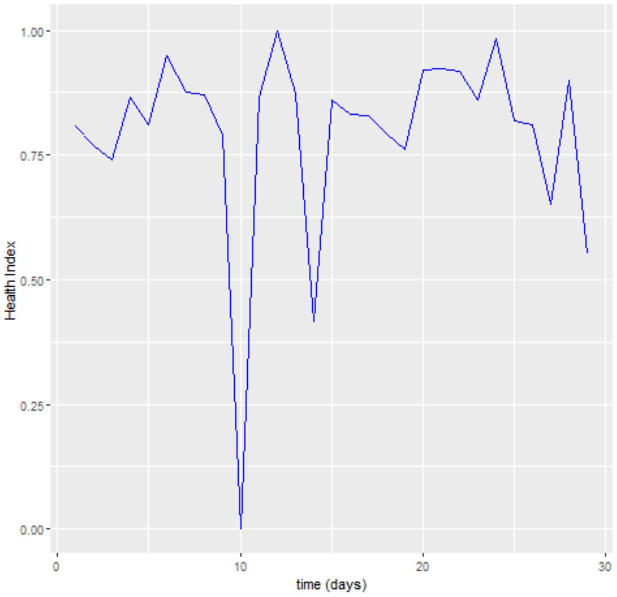
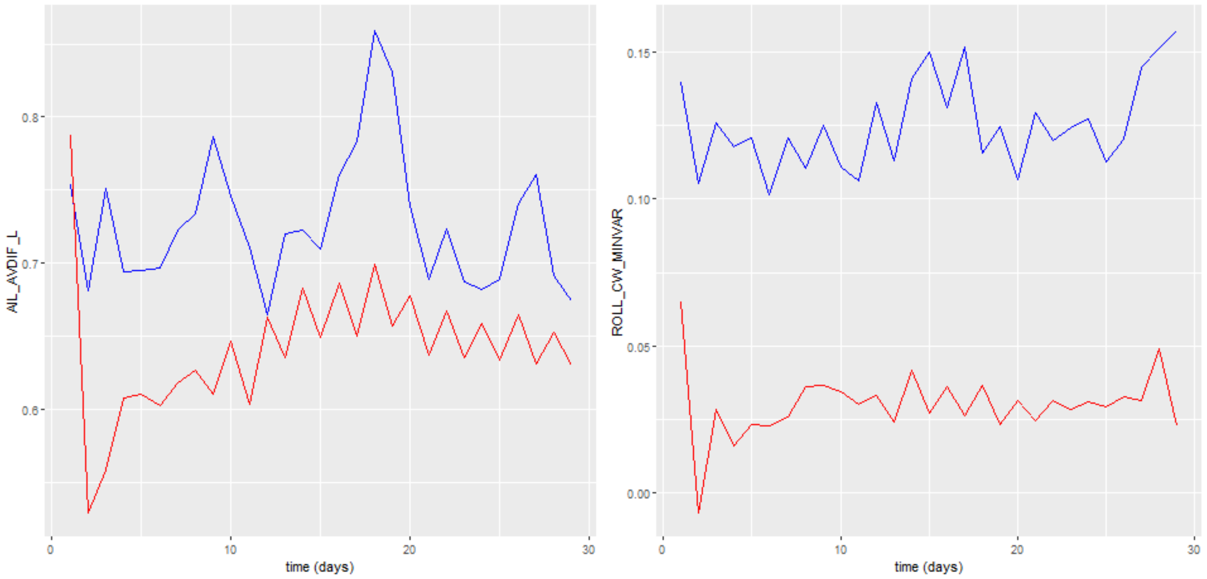


Figure 5.9: Degradation Curve for the sequence in analysis

Alike the Degradation Curve from the True Positive case, a sudden drop in health is visible, this time at the 10th day of the sequence. Once more, this was found to be due to a high count of important CMC messages in that day, in this instance concerning the Aileron, Elevator, and Electrical subsystems. It is possible that these high counts may have tipped the model into underestimating the time till the next failure.



(a)

(b)

Figure 5.10: AIL_AVDIF_L (left) and ROLL_CW_MINVAR (right) features, in blue, and their respective reconstructions, in red

In addition, inspection of the reconstructed features along with their original plots (see Fig. 5.10) revealed that the model reconstructed most features with quality, except for AIL_AVDIF_L and ROLL_CW_MINVAR, where both reconstructions fall short in value compared to the originals, seemingly already indicating problems with the Aileron subsystem, and possibly the Spoiler subsystem as well (recall that one of the Multifunction Spoilers' purposes is to aid in roll, thence responding to the movements on the control wheel), as well as problems with the Electrical and Elevator subsystems, identified from the CMC messages.

This is supported by the inspection of the maintenance reports in the period between the first point of the given sequence to the model, and the day it predicted there would be failure, plus 15 days for some margin of error. In fact, the reports show both Aileron cable complaints close to this period, and even a fault on MAU1 a mere 4 days after the predicted RUL, which was, however, labelled as a transitory condition, yet still caused a delay in the respective flight.

5.2.3 False Negative Example

The chosen sequence to represent the False Negative example was sequence number 5 from Table 5.4, with an actual RUL of 8 days, yet a predicted RUL of 105 days. This was the most extreme disparity within False Negative result instances, so an analysis was made to assess the cause of this. The sequence and an original length of 34 days, of which the algorithm was only shown the first 26.

The sequence represents a fault in the aileron cables and the corresponding Degradation Curve is shown in Fig. 5.11.

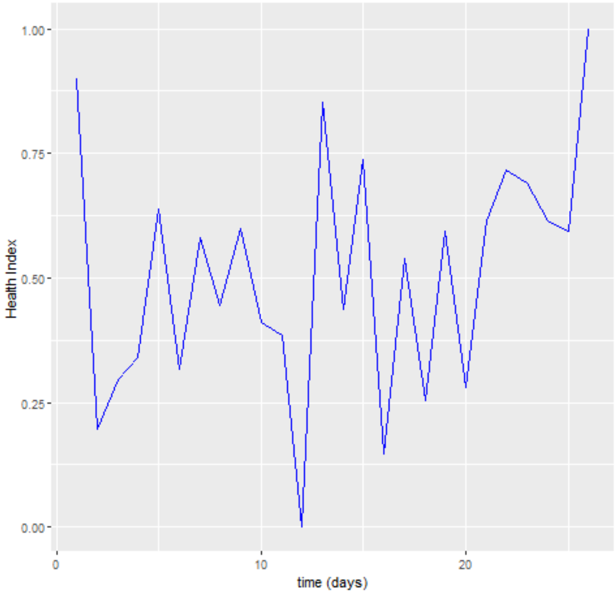
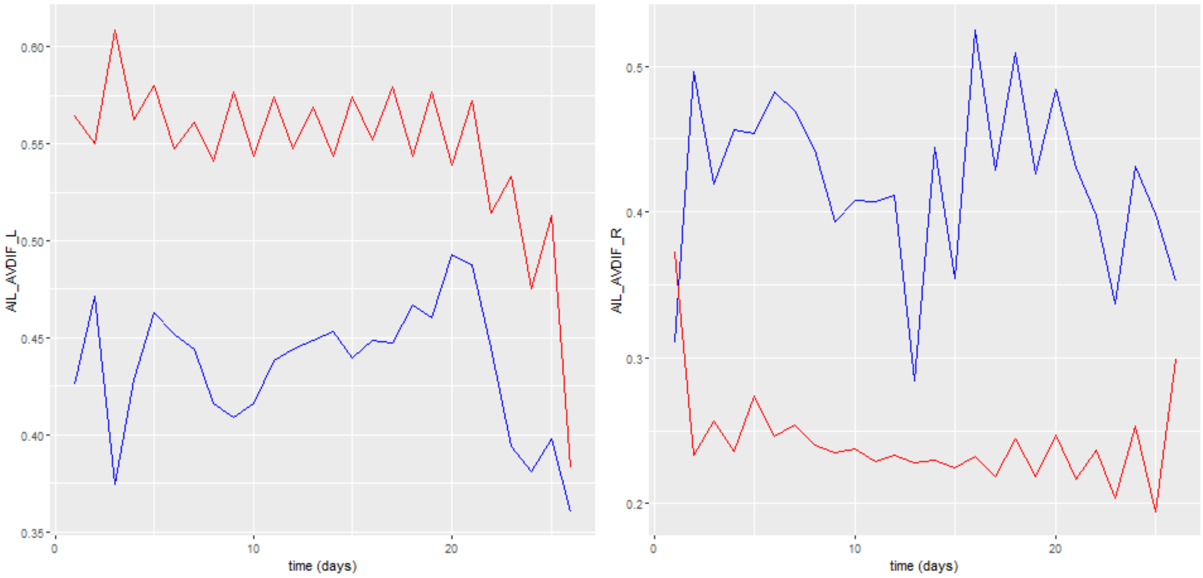


Figure 5.11: Degradation Curve for the sequence in analysis

The particularity of this curve which becomes evident is that the health of the system seems to be generally increasing over time, rather than decreasing.



(a)

(b)

Figure 5.12: AIL_AVDIF_L (left) and AIL_AVDIF_R (right) features, in blue, and their respective reconstructions, in red

Inspection of the feature reconstructions revealed that the features which were not being reconstructed up to par were the AIL_AVDIF ones (see Fig. 5.12), however, no degradation trend was visible in either of the original plots. In fact, the sequence which came before this one, chronologically, shows that these two features had in fact been on a degrading (increasing, in this case) trend which however did not carry over to the sequence being analysed, as shown in Fig. 5.13.

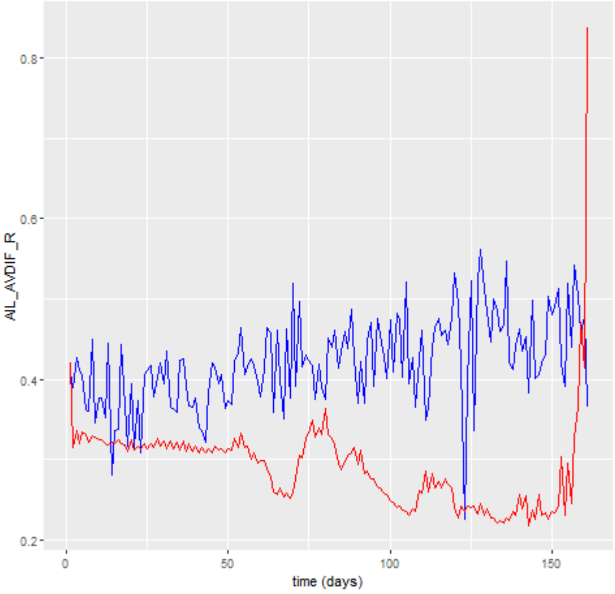


Figure 5.13: Plot of the AIL_AVDIF_R feature (blue) for the sequence immediately chronologically anterior to the one being analysed, along with its reconstruction (red)

Unlike the current sequence, the one shown in Fig. 5.13 displays a clear upward trend on the original feature, as well as a visible increasing difficulty for the model to accurately reconstruct it. This suggests that perhaps if the two sequences were to form only one sequence, it might be beneficial in predicting this fault in the Aileron subsystem. However, between them lies a multifunction spoiler fault.

5.3 Conclusions on the Model

Having analysed the RUL predictions of each set of results, it is possible to assert that filtering the system-wide peaks from the dataset — even if only partially — resulted in a substantial improvement on the accuracy of the model. Should these groups of outliers be present in the training data, they would be a great influence in the training of the autoencoder, hence perturbing the reconstructions of the sequences, causing worse results. On the other hand, should these peaks be present in a sequence, they would cause an overwhelmingly large error, thus affecting the shape of the Degradation Curve. In fact, the most prominent difference between curves from the Basic Results and curves from the System-Wide-Peak-Filtered Results is that sudden drops in health are more common, and far more prominent in the former set of predictions, than they are on the latter. This is exemplified in Fig. 5.14.

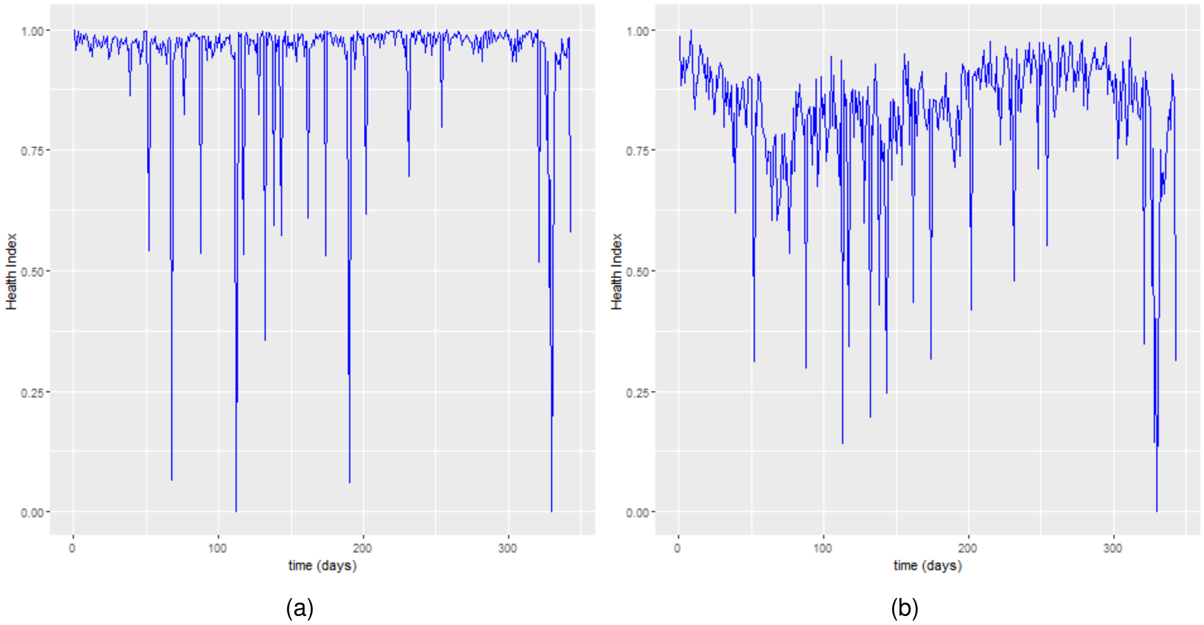


Figure 5.14: Degradation curves for the same sequence, for the model with system-wide peaks (left) and partially without system-wide peaks (right)

Another touched on subject in this chapter was the possible inadequacy of the features in representing the degradation of their respective subsystem, and thus the degradation of the overall Flight Control System. While it has been shown that certain behaviours in the feature plots seem to directly relate to some oncoming faults, in truth many other faults may not show in the gathered features, which later results in an inaccurate prediction of the RUL. As also discussed before, there is additionally the problem of feature behaviour relating to faults, yet not soon enough that this may be caught by the algorithm in time. As an example, the FLAP_MIN_EXTSPD feature represented in Fig. 3.20a could be taken into account. That plot clearly showed lower flap extend speeds very close to the vertical lines denoting the flap faults, and in that particular case, these points were visible 2 days before the fault. However, this is not always the case, and in fact, it is the exception: most times, this behavioural anomaly is only visible in the day of the fault, which as interesting as it might be from a diagnostics perspective, has no utility from a prognostics point of view.

Another posing issue possibly hindering the performance of the model was the overall quality of the data, both concerning the sensor readings, and the failure dates which were used to split the lifecycle of each aircraft. On the subject of the sensor readings, they are quite noisy, and so were the features obtained from them. While this was partly addressed by averaging all the flight summaries in each day, the issue was not completely extinguished, and the consequence are equally noisy Degradation Curves, as all figures showcasing them show. Another problem follows from there being so many possible feature behaviours for a single feature, as is the case with AIL_AVDIF_L and AIL_AVDIF_R, which may cause difficulties in the training of the autoencoder, as it is not possible to know what behaviour is normal, and what is anomalous. Additionally, sometimes certain values or behaviour evolutions seemingly lead to a fault, yet the same values do not lead to a fault at some other period of the life of that aircraft, or on some other aircraft.

As for problems with the failure data, they stem from two sources. On the one hand, the reports from which the faults were gathered display a certain issuing date and issuing time, but these are only accurate if the report was input in the system at the exact flight the issue took place, which is usually not the case. When switching from flights to days as the definition of cycle, this issue becomes moot however, as the reports could be with pinpoint accuracy, and still in the model it was assumed that the fault took place at the end of the day, which is to say after the last flight of the day. Despite this, the most pressing matter concerning fault data is still likely the subjectivity involved in reading and adding the reports of faults which were not FLT CTRL NO DISPATCH. As already discussed, there were too many reports, and a first approach was to add only reports which mentioned the replacement of a component. Simply that already adds a certain level of interpretation required: should all of these reports be added as faults regardless of whether or not the features show changes in their behaviours accordingly?; should scheduled inspection actions that also require replacements be added as well? And this is all beside the fact that after reviewing all these reports, abnormal behaviours were still searched for in the features so as to further look for other reports which, while not mentioning replacements, might have caused that change. This also adds subjectivity, as it depends on the eye of who is looking at the feature plots.

And finally, likely the major issue with this approach, was that too many subsystems which were mostly independent from each other were being modelled together. There were too many failure modes, so even if the algorithm might be able to predict a fault on the Electrical System for instance, that was no guarantee that it is capable of predicting a fault on the Aileron System. This would have been of interest to explore, by separating all systems and analysing them one by one, with only faults and features pertaining to the respective system, and perhaps the Electrical System being modelled together with each of the subsystems which depend on it. However, the lack of failure data, which had already been an issue with all the systems together, would now completely impede such study: there was no single system with enough failures to create an entire dataset of them, and the only system coming close to that possibility was the Electrical System, which would invariably require the analysis of features from more subsystems along with it anyway.

To conclude, it seems that most obstacles revolved around the lack of failure data, which is evidently good for Portugália Airlines, but it hindered the model. The obtained results from the validation sets and the test sets being so different from one another and the almost sporadic estimates on some of the samples seem to imply that these predictions stand for themselves and do not guarantee that the model would perform with the same overall quality on another test set. Still, the analysis done on the second set of results suggests that there is validity in the model's interpretation in what is an abnormal behaviour, so a RUL estimation corresponding to a small value, while not necessarily meaning there is an imminent fault, may be an indicative that the aircraft may benefit from a brief inspection.

Chapter 6

Conclusion

The ultimate objective of this work was to provide a solution in failure prediction on the Flight Control System to allow for a better maintenance strategy beyond scheduled preventive maintenance, and run-to-failure maintenance, and in that front, being able to predict a few faults is still better than not predicting any, so a solution which was not prone to outputting False Positives was pursued, so that whenever a small RUL was predicted, the action should be to inspect the feature reconstructions and if possible identify the source of the problem, or to directly inspect the aircraft.

The following sections detail this work's achievements, obtained when procuring this solution, along with any future work which might be in order.

6.1 Achievements

This work arrived at the objective of predicting the Remaining Useful Life of a set of test sequences of flights, with reasonable accuracy in predictions of up to 20 days, and granted that the number of flights to assess is great enough so as to give the model enough data points to make its prediction.

Additionally, many statistics to gauge the Flight Control System's health were created, for each of its many subsystems, and many of them proving to directly relate to faults or system degradation. While features such as AIL_AVDIF may have too much variability from many factors to conclude on the subsystem's degradation, they may still be useful to assess the performance of the controls, and to cross check complaints with (features measuring forces may equally be useful for cross checking with complaints from the pilots). Other features such as ROLL_CW_MINVAR and SPOILS_MINVAR were proven to relate to some faults, and seeing as sometimes a maintenance intervention worsens their respective behaviours, it may be practical to keep updated plots of each to catch any such changes in behaviour almost as soon as they happen. This is also valid for many other features, as many were seen correlating to some faults, even if only one or two.

Finally, these features allowed the identification of problems in either the recording system, or some software glitch, which, while not problematic in the aircraft's operations, may be a hindrance in further analysis of aircraft flight data.

6.2 Future Work

As has been discussed before, the main problem faced in this project was lack of failure data. It is in every way interesting to pick the algorithm back up in some years time, when more FLT CTRL NO DISPATCH faults have been accumulated, to train and test it again, with faults exclusively from that source. Additionally, as discussed before, performing separate analyses on the different subsystems in separate is also a route worth committing to with enough failure data. Should this be done, it would be possible to directly infer which system was at fault when a certain RUL is estimated.

Alternatively, a more in depth analysis and gathering of the maintenance reports could also better the performance of the model, although this would be quite manifestly time consuming.

Additionally, in order to also try to circumvent the lack of failure data, the model could be trained with all available data (i.e. not only this work's train set, but also test and validation sets), then it could be implemented in Portugália Airline's environment to see how close its predictions are to future faults on the Flight Controls System.

Finally, implementation of the feature gathering routines in Portugália Airlines' environment could prove to be beneficial for the reasons already mentioned in the above section, and even if not for their predictive capabilities, then for their troubleshooting ones. As has already been shown, many features do relate to faults, and some of them only show signs of anomaly much too close to the day of the fault for their predictive capabilities to be of any worth. Many a time, troubleshooting the Flight Control System is not as simple as reading an error code and cross checking it with the maintenance manuals. Perhaps with the aid of these variables, troubleshooting becomes easier and hence more efficient, therefore avoiding, or otherwise minimising, flight delays. The program would then not only work as a Predictive Maintenance solution, but also a Health Monitoring one.

Bibliography

[1] AIRBUS, *Global Market Forecast 2019-2038*. AIRBUS S.A.S, August 2019.

[2] IATA, *Airline Maintenance Cost Executive Commentary*. International Air Transport Association, December 2019.

[3] R. Heisey, “Low maintenance costs and high dispatch reliability,” *Boeing Aero Magazine*, no. 17, July 2002.

[4] Infraspeak, “Reactive maintenance: Definition, advantages and examples.” `https://blog.infraspeak.com/corrective-maintenance/`, 2020. [Online; Accessed 18 Dec. 2020].

[5] T. Hoyland, C. Spafford, and A. Medland, *MRO Big Data – A Lion or a Lamb?* Innovation and Adoption in aviation MRO, Oliver Wyman, 2016.

[6] AIRBUS, “FAST – flight airworthiness support technology,” *Airbus Technical Magazine*, no. 58, August 2016.

[7] Boeing, “Airplane health management,” *Real-time Operations Airplane Health Management*.

[8] K. Bhattacharya, J.-P. Cresci, and P. Lortie, *Machine Learning: A Turning Point for Predictive Maintenance?* VELOCITY, Oliver Wyman, 2017.

[9] Embraer, “E-jets. the business warrior.” `https://www.embraercommercialaviation.com/fleet/e-jets/`. [Online; Accessed 14 Dec. 2020].

[10] Embraer, “Portugália airlines fleet.” `https://www.portugalia-airlines.pt/Pages/Fleet.aspx`. [Online; Accessed 17 Dec. 2020].

[11] Embraer, *Maintenance Training Manual*. Embraer ERJ-170/190 (GE CF34) General Familiarization, Embraer S.A.

[12] Embraer, *Aircraft Maintenance Manual*, ch. System Description Section. Embraer S.A., August 2008.

[13] A. Unlimited, “Ata 100 chapters complete list.” `https://www.aerospaceunlimited.com/ata-chapters/`. [Online; Accessed 18 Dec. 2020].

[14] E. A. Solutions, “Ata chapters.” https://www.exsyn.com/wiki/ata-chapter. [Online; Accessed 17 Dec. 2020].

[15] E. White, “Flap slat accessory module and flap skew detection system,” *Boeing Aero Magazine*, no. 6, 1999.

[16] D. Soni, “Supervised vs. unsupervised learning.” https://towardsdatascience.com/supervised-vs-unsupervised-learning-14f68e32ea8d, Mar. 2018. [Online; Accessed 14 Dec. 2020].

[17] A. Bisht, “Classification vs. regression.” https://www.geeksforgeeks.org/ml-classification-vs-regression/, Dec. 2019. [Online; Accessed 14 Dec. 2020].

[18] M. Phi, “Illustrated guide to lstm’s and gru’s: A step by step explanation.” https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21, Sep. 2018. [Online; Accessed 27 Jun. 2020].

[19] P. Malhotra, V. TV, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, “Multi-sensor prognostics using an unsupervised health index based on lstm encoder-decoder.” arXiv 1608.06154, 2016.

[20] N. Gugulothu, V. TV, P. Malhotra, L. Vig, P. Agarwal, and G. Shroff, “Predicting remaining useful life using time series embeddings based on recurrent neural networks.” arXiv 1709.01073, 2017.

[21] J. Sublime and E. Kalinicheva, “Automatic post-disaster damage mapping using deep-learning techniques for change detection: Case study of the tohoku tsunami,” *Remote Sensing*, vol. 11, no. 9, 2019.

[22] T. Wang, *Trajectory Similarity Based Prediction for Remaining Useful Life Estimation*. PhD thesis, University of Cincinnati, 2010.

[23] NASA, “Pcoe datasets.” https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/. [Online; Accessed 2 May 2020].

[24] X. Zhang *et al.*, “Normalize data.” https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/normalize-data, Jun. 2019. [Online; Accessed 12 Dec. 2020].

Appendix A

List of Created Sensor Features

The following are the sensor features which were created for the analysis of the Flight Control System.

A.1 Electrical System

By inspection of the many flights of the various aircraft, it was possible to conclude that, for each subsystem with dedicated ACEs, the number of engaged ACE channels at a time is mostly the same, and exceptions seem to happen very sporadically and many times at the very beginning of the flight (in the taxi-out phase). While this may be simply due to the AGS software's mis-categorisation of the flight phase (i.e. maybe the taxi-out phase actually started some seconds before or after the software's acknowledgement), it could also be due to a delay in engaging the ACEs. Moreover, when these exceptions did not happen in the taxi-out phase it could be due to some other problem in the electrical system. So the features created concerning this are variables which count how many times the number of engaged ACE channels in each subsystem is not the respective usual value:

- FSACE: Number of times that the number of engaged FS-ACE channels is not 6 in a flight (starting from the Taxi-Out flight phase);
- HSACE: Number of times that the number of engaged HS-ACE channels is not 1;
- ELACE: Number of times that the number of engaged P-ACE elevator channels is not 4;
- RDACE: Number of times that the number of engaged P-ACE rudder channels is not 2.

Furthermore, with the exception of the Flap/Slat subsystem where all the possible FS-ACE channels are engaged at the same time, there are spare channels in every subsystem. So, for each flight, only a portion of the total number of ACE channels is used, and upon inspection of all flights it was possible to conclude that the groups of engaged/disengaged ACE channels are typically well defined and are interchanged each day. To exemplify, there are eight FCM channels for the elevator P-ACEs, corresponding to 1A, 1B, 2A, 2B, 3A, 3B, 4A and 4B, however only four are used each flight, and there

are only two possible combinations: in a single flight, either channels 1A, 2B, 3A and 4B are used, or channels 1B, 2A, 3B and 4A, with exceptions happening extremely sporadically.
Additionally, it was found that the accuracy of some sensor readings depends on which group of channels is used for each subsystem, and because of this, it was thought useful to know which channels are most active in each flight:

- ELACE_ENG: Which group of P-ACE elevator channels is engaged in the flight;
- HSACE_ENG: Which HS-ACE channel is engaged in the flight;
- RDACE_ENG which group of P-ACE rudder channels is engaged in the flight.

A.2 Ailerons

The aileron subsystem has two torque tubes, each having three bellcranks connected to position sensors, so that there are six readings total for the control wheel position. It was thought that the level of disparity between these readings might correlate to faults in the Flight Control system, so features were created to account for that:

- ROLL_CW_AVVAR: Average variance between all of the control wheel roll position sensor readings;
- ROLL_CW_MAXVAR: Maximum variance between all of the control wheel roll position sensor readings;
- ROLL_CW_MINVAR: Minimum variance between all of the control wheel roll position sensor readings.

Embraer provides a table of correspondence between control wheel input and aileron surface deflection, so it was possible to assess how close, or how far, the real deflections were from the theoretical ones, given the pilots' input. The average, minimum, and maximum differences recorded in each flight were then taken as features:

- AIL_AVDIF_L: Average difference between theoretical and actual left aileron deflection given control wheel input;
- AIL_AVDIF_R: Average difference between theoretical and actual right aileron deflection given control wheel input;
- AIL_MAXDIF_L: Maximum difference between theoretical and actual left aileron deflection given control wheel input;
- AIL_MAXDIF_R: Maximum difference between theoretical and actual right aileron deflection given control wheel input;

- AIL_MINDIF_L: Minimum difference between theoretical and actual left aileron deflection given control wheel input;
- AIL_MINDIF_R: Minimum difference between theoretical and actual right aileron deflection given control wheel input;

Finally, the following features were created to try and see if there might be a correlation between possible looseness or stiffness of the control wheel (even if only noticeable in the data) and system degradation:

- AIL_AVFC: Average force applied to the control wheel;
- AIL_MAXFC: Maximum force applied to the control wheel;
- AIL_MINFC: Minimum force applied to the control wheel.

A.3 Rudder

There are two sensor readings for the rudder pedal position. Once again, it was thought that the evolving difference between the two readings might correlate to degradation, so the following features were created:

- RUD_PDL_AVDIF: Average difference between rudder pedal left and right position sensor readings;
- RUD_PDL_MAXDIF: Maximum difference between rudder pedal left and right position sensor readings;
- RUD_PDL_MINDIF: Minimum difference between rudder pedal left and right position sensor readings.

While there is not a table of correspondence between pedal input and rudder surface deflection, like there was for the control wheel and the ailerons, it was still possible to evaluate the system on an input/output basis. A group of flights considered "healthy" for each aircraft was taken and a linear regression was applied to create a theoretical model of the behaviour of this control surface. As such, and much like the ailerons, features were created to compare this theoretical deflection with the actual deflection, given pilots' pedal positions at every instant of the flight.

- RUD_AVDIF: Average difference between theoretical and actual rudder surface position;
- RUD_MAXVDIF: Maximum difference between theoretical and actual rudder surface position;
- RUD_MINDIF: Minimum difference between theoretical and actual rudder surface position.

The rudder surface can be moved by either the upper or the lower PCU, each having its own position sensor, all the while the rudder being considered a rigid body. In that sense, the difference between the two sensor readings was also studied via the following features:

- RUDS_AVDIF: Average difference between upper and lower rudder sensors;
- RUDS_MAXDIF: Maximum difference between upper and lower rudder sensors;
- RUDS_MINDIF: Minimum difference between upper and lower rudder sensors.

Finally, the necessary force exerted on the pedals was also studied:

- RUD_PDL_AVFC: Average force applied to the rudder pedals;
- RUD_PDL_MAXFC: Maximum force applied to the rudder pedals;
- RUD_PDL_MINFC: Minimum force applied to the rudder pedals.

A.4 Elevator

There are two available readings for control column position: one for the pilot, and one for the copilot. While both the control columns are mechanically connected, this means that both sensors should read similar values, so the difference between the two was taken as a feature, to allow further study:

- PITCH_POS_AVDIF: Average difference between position readings of pilot and copilot control columns;
- PITCH_POS_MAXDIF: Maximum difference between position readings of pilot and copilot control columns;
- PITCH_POS_MINDIF: Minimum difference between position readings of pilot and copilot control columns.

Like the rudder, given the fact that there is not a table available with control column position versus elevator surface deflection provided by Embraer, a set of flights considered “healthy” was used to compute a linear regression model to take as theoretical input/output relationship. With this, a set of features comparing the theoretical surface deflection with the real surface deflection, given pilot input, was created:

- ELEV_AVDIF_L Average difference between theoretical and actual left elevator deflection given pilot input;
- ELEV_AVDIF_R Average difference between theoretical and actual right elevator deflection given pilot input;
- ELEV_MAXDIF_L Maximum difference between theoretical and actual left elevator deflection given pilot input;
- ELEV_MAXDIF_R Maximum difference between theoretical and actual right elevator deflection given pilot input;

- ELEV_MINDIF_L Minimum difference between theoretical and actual left elevator deflection given pilot input;
- ELEV_MINDIF_R Minimum difference between theoretical and actual right elevator deflection given pilot input.

Each elevator surface can be controlled by either of two PCUs (inboard or outboard PCU), so there are two position sensors per elevator surface. Both of the sensors are connected to different P-ACEs, which in turn connect to different FCMs, so a possible difference between the two readings could correlate to degradation or a fault within this circuit. As such, features were created to account for that:

- ELEVS_AVDIF_L: Average difference between elevator left inboard and outboard sensor readings;
- ELEVS_AVDIF_R: Average difference between elevator right inboard and outboard sensor readings;
- ELEVS_MAXDIF_L: Maximum difference between elevator left inboard and outboard sensor readings;
- ELEVS_MAXDIF_R: Maximum difference between elevator right inboard and outboard sensor readings;
- ELEVS_MINDIF_L: Minimum difference between elevator left inboard and outboard sensor readings;
- ELEVS_MINDIF_R: Minimum difference between elevator right inboard and outboard sensor readings.

Finally, the average and maximum force applied to the control column in each flight were taken to find correlations between system degradation and possible looseness or stiffness of the control.

- ELEV_AVFC: Average force applied to the control column;
- ELEV_MAXFC: Maximum force applied to the control column;
- ELEV_MINFC: Minimum force applied to the control column.

A.5 Flaps

Like the ailerons, Embraer provided a table of correspondence between pilot input on the controls, and surface deflection. Furthermore, the values for both flap deflection and Slat/Flap Lever position are discrete. As such, the flaps would always be reported by the software to eventually arrive at the exact theoretical deflection, so a direct measure of either delay, or speed to arrive at that deflection could be computed:

- FLAP_AV_EXTSPD: Average flap mean angular extend speed;

- FLAP_MAX_EXTSPD: Average flap maximum angular extend speed;
- FLAP_MIN_EXTSPD: Average flap minimum angular extend speed;
- FLAP_AV_RETSPD: Average flap mean angular retract speed;
- FLAP_MAX_RETSPD: Average flap maximum angular retract speed;
- FLAP_MIN_RETSPD: Average flap minimum angular retract speed.

However, characteristics of some flights made it so the above-mentioned features were prone to missing values, so an extra feature was created, which was impervious to this:

- FLAP_TIMEFRACTION: Fraction of total time in a flight where the flaps are reported to be in the wrong deflection, given lever input, considering an ideal flap response with no delay, and disregarding the climb and the cruise flight phases, where the flaps are not used.

A.6 Spoilers and Speed Brakes

There are three possible readings for the speed brake lever position, so it was thought useful to gauge the disparity level between them, so features accounting for the variance between these readings were created:

- SPD_BRK_AVVAR: Average variance between all of the speed break lever position indicators;
- SPD_BRK_MAXVAR: Maximum variance between all of the speed break lever position indicators;
- SPD_BRK_MINVAR: Minimum variance between all of the speed break lever position indicators;

For the Spoilers and Speed Brakes, once again an empirical model for pilot input versus surface deflection was obtained from flights assumed healthy. Given the lever position, throughout the flight, the deflections were calculated based on this model, and the average, maximum and minimum differences between these values and the actual recorded ones were stored:

- SPOIL_AVDIF: Average difference between theoretical and actual spoiler surfaces positions;
- SPOIL_MAXDIF: Maximum difference between theoretical and actual spoiler surfaces positions;
- SPOIL_MINDIF: Minimum difference between theoretical and actual spoiler surfaces positions.

Furthermore, in order to study the possible correlation between the disparity of all six position readings for the multifunction spoilers when the speed brake lever is used (which should result in a symmetrical and equal deflection in all spoilers), the following features were also created:

- SPOIL_AVVAR: Average variance between all of multifunction spoiler position readings;
- SPOIL_MAXVAR: Maximum variance between all of multifunction spoiler position readings;
- SPOIL_MINVAR: Minimum variance between all of multifunction spoiler position readings;

A.7 Slats

Akin to the flaps, there is a correspondence between the flap/slat lever position and the slat surface deflection. The Slat subsystem was given the same treatment as the Flap subsystem, as they are both quite similar:

- SLAT_AV_EXTSPD: Average slat mean angular extend speed;
- SLAT_MAX_EXTSPD: Average slat maximum angular extend speed;
- SLAT_MIN_EXTSPD: Average slat minimum angular extend speed;
- SLAT_AV_RETSPD: Average slat mean angular retract speed;
- SLAT_MAX_RETSPD: Average slat maximum angular retract speed;
- SLAT_MIN_RETSPD: Average slat minimum angular retract speed;
- SLAT_TIMEFRACTION: Fraction of total time in a flight where the slats are reported to be in the wrong deflection, given lever input, considering an ideal slat response with no delay, and disregarding the climb and the cruise flight phases, where the slats are not used.

Appendix B

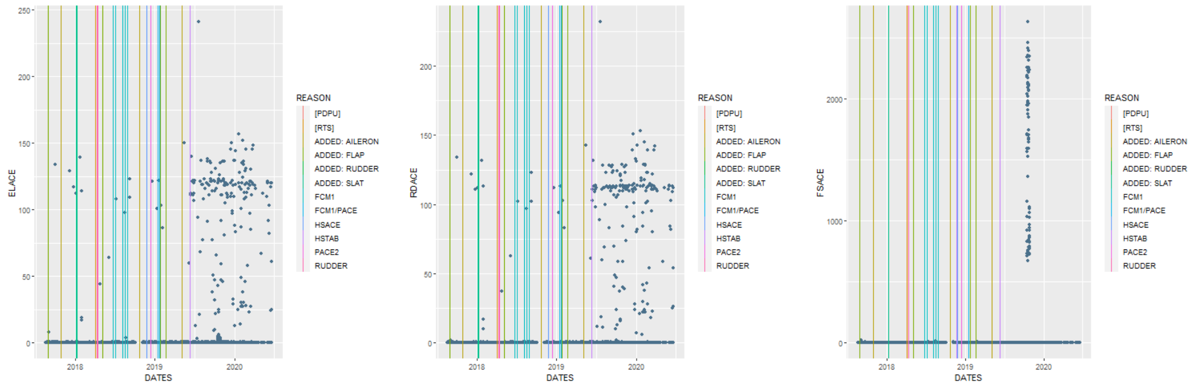
Feature Selection/Feature Plot Visual Inspection — Exhaustive List

The total created features amounted to 75, yet of these only a selection would later be used in the algorithm, since there were many which would only add to the run time of the algorithm and nothing more. This selection should be made based on correlation to the faults. This was done via a visual inspection of the plots of each feature throughout time (i.e. the dates of the flights) to assess their usefulness in predicting impending faults. This section covers that analysis.

It should be noted that in testing it was found that some sensor readings were susceptible to dramatic outliers both due to the rapid command movements in the taxi-out phase, and the inherent noise the signal carries. In these cases, the respective features which represented an average were found to be more useful when they were used as a median, so they were changed.

The following shows the most prominent plot examples for each feature type, and is an exhaustive list of the reasoning behind the discarding/keeping of all created features, which includes both what has been said in the main text and the rest of the features which had not been addressed there.

B.1 Electrical System



(a) (b) (c)

Figure B.1: Plots for the ELACE (left), RDACE (centre), and FSACE (right) features on a single aircraft

ELACE is, for most aircraft, very sparse, presenting mostly values of zero, and with outliers which do not seem to have any correlation to the faults. In fact, for aircraft where this feature is not so sparse, the distributions of the points seem to bear little to no information about fault imminence. As presented in Fig. B.1a, there start being more ELACE occurrences mid 2019, and this behaviour continues for at least a year, with no reported issues, which seems to indicate this is not a good candidate for prognostics analysis.

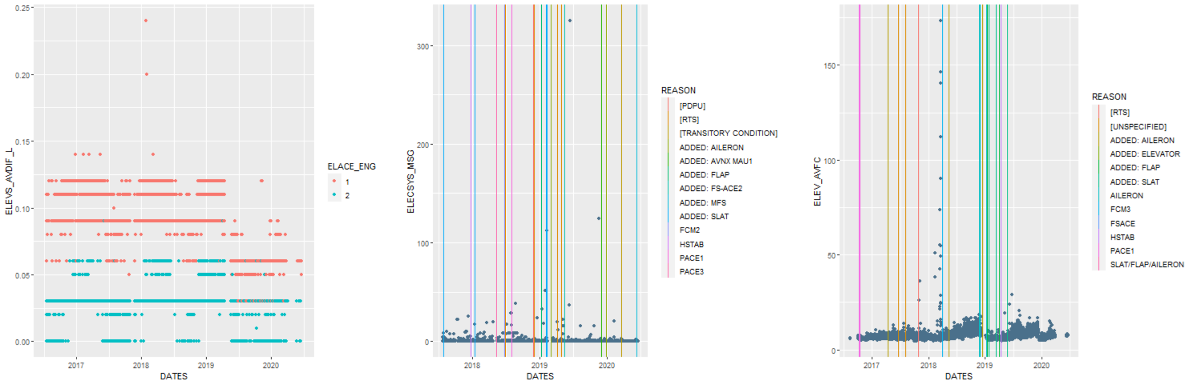
RDACE mostly presents a distribution similar to ELACE, which is not unexpected given that both of these features pertain to P-ACEs. As such, this feature carries the same characteristics as ELACE.

FSACE, like ELACE, is very sparse and its occurrences seem to have no correlation to faults on either the Electrical, Flap, or Slat subsystems. However, and as shown in Fig. B.1c, it carries a characteristic which will henceforth be referred to as “system-wide peaks”. In this figure, one of these very distinct peaks is visible in the last quarter of 2019 — these are phenomena of anomalous behaviour which, when present at a given period, show in many other features, across multiple subsystems. These abnormalities will be discussed later in this section when analysing the features directly pertaining to the Flap and Slat subsystems.

HSACE was found to always be a constant of zero, therefore carrying no information for the algorithm.

All of these characteristics dictated that these features be discarded, as they were not worth the additional computing time required for having them.

As for the ELACE_ENG, HSACE_ENG, and RDACE_ENG features, their initial function was not to point to system degradation (as they only show which groups of ACEs were chosen for the specific flights), but rather to aid the algorithm in identifying and better dealing with changes in behaviour which are not necessarily due to an impending fault, and instead are due to this change in ACEs, as is shown in Fig. B.2a, where the ELACE_ENG value is represented by the colour of each point in the plot. This would be advantageous since the measurement of health will be taken from the error of the AE in reconstructing the data it is given. Of these, only ELACE_ENG was kept, as RDACE_ENG largely coincides with it, and HSACE_ENG pertains to the horizontal stabiliser, for which no features were created.



(a) Example of feature dependency on the ELACE_ENG variable

(b) Plot for the ELECSYS_MSG feature for a given aircraft

(c) Example of a fault on the Electrical System's influence on features belonging to other subsystems

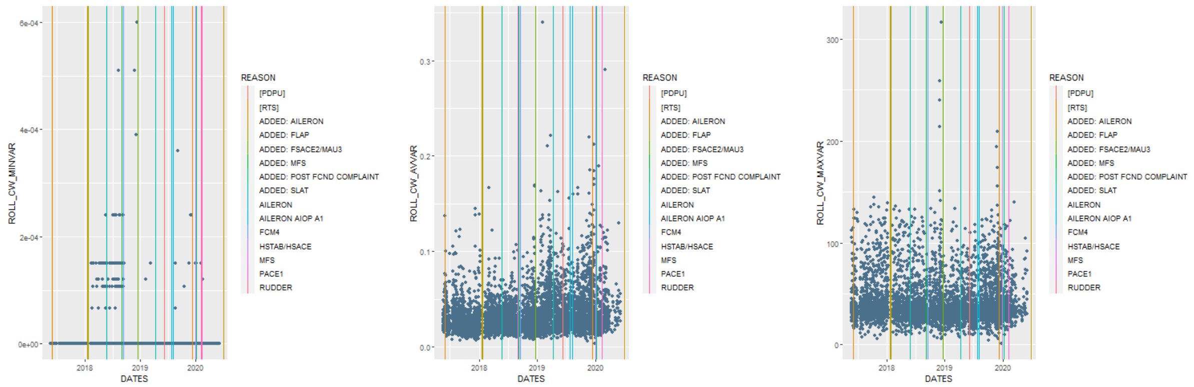
Figure B.2: Plots representing some characteristics of the Electrical System

Finally, the ELECSYS_MSG feature shows that its distribution might be related to certain faults not limited to the electrical system. In Fig. B.2b, for example, the relatively high incidence of outliers around the first half of 2019 is concurrent with the high fault frequency in the same period. In fact, apart from two other outliers, this period ends with a fault, and respective maintenance action, on MAU1.

Given these examples across many aircraft, this feature was kept. It should be noted that it was decided all features regarding the FHDB messages were to be kept, given the importance of these messages and how many of them dictated the actions of the maintenance team.

Furthermore, while this means that there was only one feature directly representative of the Electrical System's degradation state, this should not be taken as a sign of its unimportance. In fact, the most common source of faults in the dataset is precisely this subsystem, so it should be handled as such — and despite this apparent issue, it is in truth not being belittled. Verily, being as global a system as it is, it affects all others, and hence many anomalies in the behaviours of features belonging to other systems (such as the Rudder, or Elevators) can be traced back to faults in the Electrical System. An example of this is given by Fig. B.2c, in the first quarter of 2018, where an unusual amount of force seems to be needed to move the elevator control column in some flights; this issue eventually ends after a fault and its respective corrective action on FCM1 — an Electrical System component. As a final note on this matter, although it might seem so, the described behaviour in Fig. B.2c is not a system-wide peak, as it is exclusive to this feature at this period for the given aircraft.

B.2 Ailerons



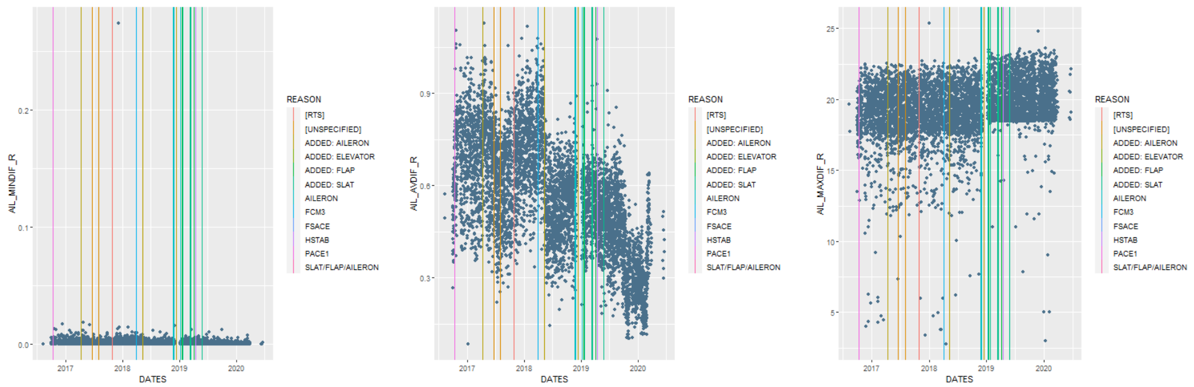
(a)

(b)

(c)

Figure B.3: Plots for the ROLL_CW_MINVAR (left), ROLL_CW_AVVAR (centre), and ROLL_CW_MAXVAR (right) features on a single aircraft

Of the three ROLL_CW_xxVAR features (presented in Fig. B.3 for a specific aircraft), only ROLL_CW_MINVAR was kept, for two reasons. The first being its clarity in possible fault correlation, which is visible in Fig. B.3a as a behaviour change by the second half of 2018 extending to the first quarter of 2019, which worsens by the end of 2018, and eventually ends at an FCM4 fault. The second reason was for this feature generally being, of the three, the one least affected by system-wide peaks (as will be shown later in this section that these peaks come in detriment to the data, rather than improvement).



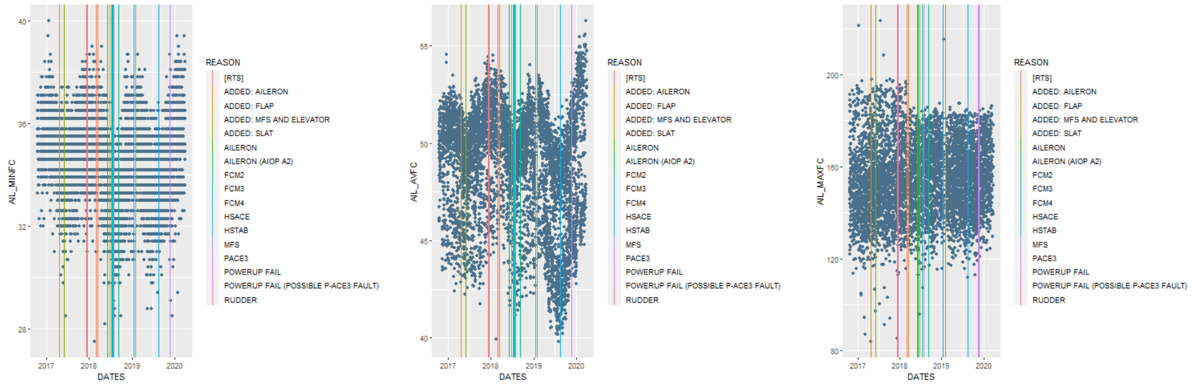
(a)

(b)

(c)

Figure B.4: Plots for the AIL_MINDIF_R (left), AIL_AVDIF_R (centre), and AIL_MAXDIF_R (right) features on a single aircraft

Regarding the set of features presented in Fig. B.4, along with their left side counterparts, only AIL_AVDIF_R and AIL_AVDIF_L were kept, seeing as they presented the greatest variability with which coming visible behaviour changes after maintenance actions. Fault correlation with these features has already been discussed in the paragraphs accompanying **??** and **??**, in Section 3.3.2 - Lifecycle Splitting so this is not repeated here. An additional reason these features were chosen over the rest was their imperviousness to system-wide peaks.



(a)

(b)

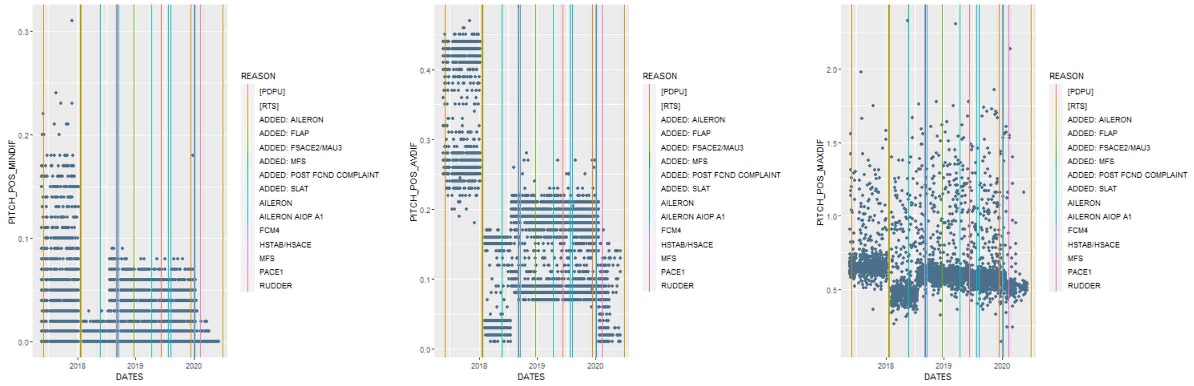
(c)

Figure B.5: Plots for the AIL_MINFC (left), AIL_AVFC (centre), and AIL_MAXFC (right) features on a single aircraft

Finally, for the aileron force features (Fig. B.5), both AIL_MINFC and AIL_AVFC display seasonal behaviours which are only disrupted by maintenance actions on the system (see **??** for an example of such disruptions). According to these two features, generally, more force is required to move the aileron controls in the winter season, and conversely, less is required during the summer season.

As for AIL_MAXFC, it is the only aileron force feature independent from season, and it is also in fact the one where the majority of complaints on control stiffness is visible, as discussed before, in Section 3.3.2 - Lifecycle Splitting. For this reason, it was the only one which was kept.

B.3 Elevators



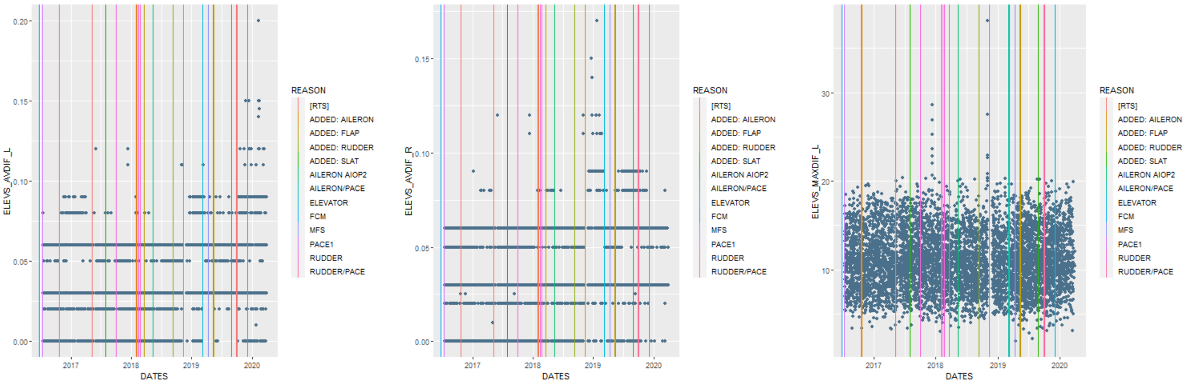
(a)

(b)

(c)

Figure B.6: Plots for the PITCH_POS_MINDIF (left), PITCH_POS_AVDIF (centre), and PITCH_POS_MAXDIF (right) features on a single aircraft

PITCH_POS_AVDIF, PITCH_POS_MAXDIF, and PITCH_POS_MINDIF (see Fig. B.6) all have quite stable behaviours which only change following a modification to the elevator control columns, most commonly during maintenance checks. Moreover, they present no visible correlations to elevator faults and all share the same general plot characteristics, so they were all discarded.



(a) (b) (c)

Figure B.7: Plots for the ELEVS_AVDIF_L (left), ELEVS_AVDIF_R (centre), and ELEVS_MAXDIF_L (right) features on a single aircraft

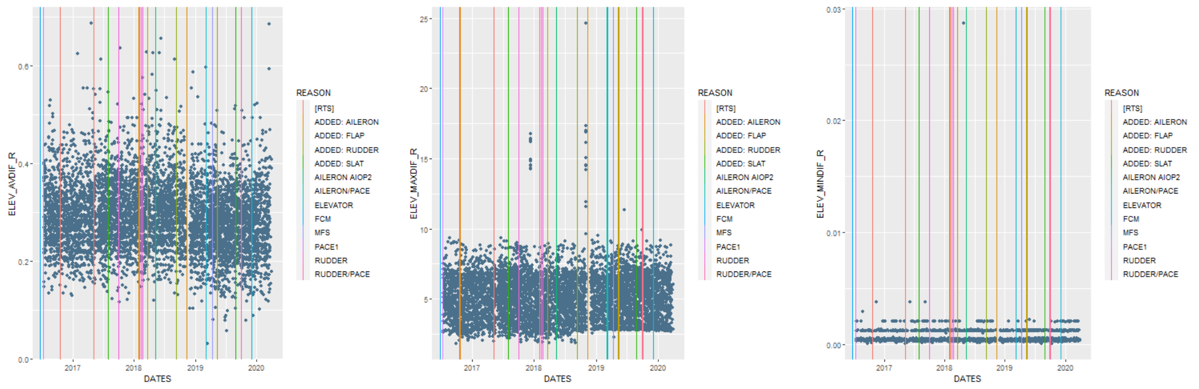
The ELEVS_AVDIF features have been identified as correlating to some faults in the dataset. The plots presented in Fig. B.7a and Fig. B.7b show an interesting example regarding this. Specifically, for the right-side elevator surface, Fig. B.7b, it shows that the difference between readings of the inboard and outboard sensors is seldom greater than 0.1° in magnitude, and when it is, the same happens for the left-side elevator sensors. However, this is not the case for the period starting from the end of 2018, where it is visible for the right side an increase in sensor reading discrepancy, which only ends after a fault in the elevator subsystem, which was acted upon by replacing the right inboard elevator sensor.

Moreover, on the left side variable, an increase in discrepancy is also visible by the end of 2019, and it would not be illegitimate to think that there might be an impending fault concerning either of the left side sensors, however by this time the data is considerably more scarce due to there being less flights.

Given this, both ELEVS_AVDIF_L and ELEVS_AVDIF_R were kept in the dataset.

Regarding the ELEVS_MAXDIF_L and ELEVS_MAXDIF_R features (ELEVS_MAXDIF_L being represented in Fig. B.7c for a specific aircraft), they are both random with unchanging behaviour across all aircraft safe for the system-wide peaks, which means they do not carry any additional information, so they were discarded.

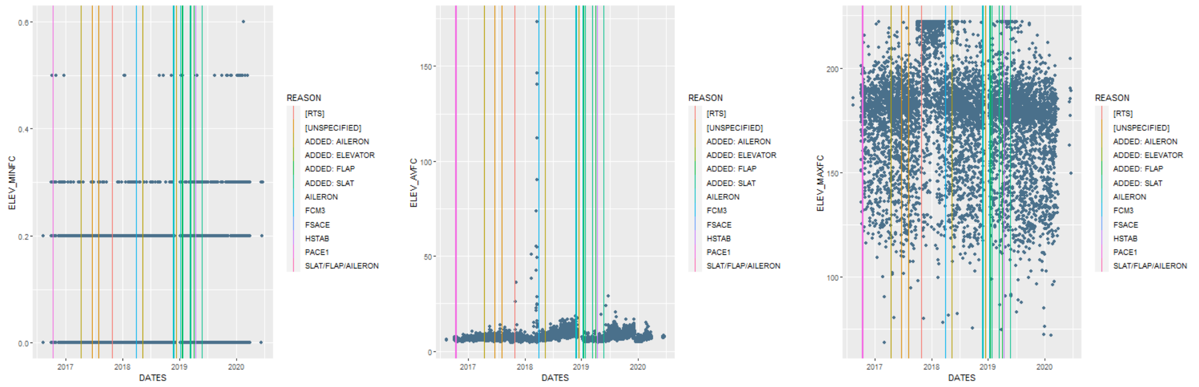
The ELEVS_MINDIF features were both found to be a constant of zero in all aircraft, so they were also removed from the dataset.



(a) (b) (c)

Figure B.8: Plots for the ELEV_AVDIF_R (left), ELEV_MAXDIF_R (centre), and ELEV_MINDIF_R (right) features on a single aircraft

All of the features represented in Fig. B.8, along with their left side counterparts, act fairly randomly in most aircraft, and no evident correlations to faults of the elevator or the electrical systems are visible. These features were therefore all discarded.



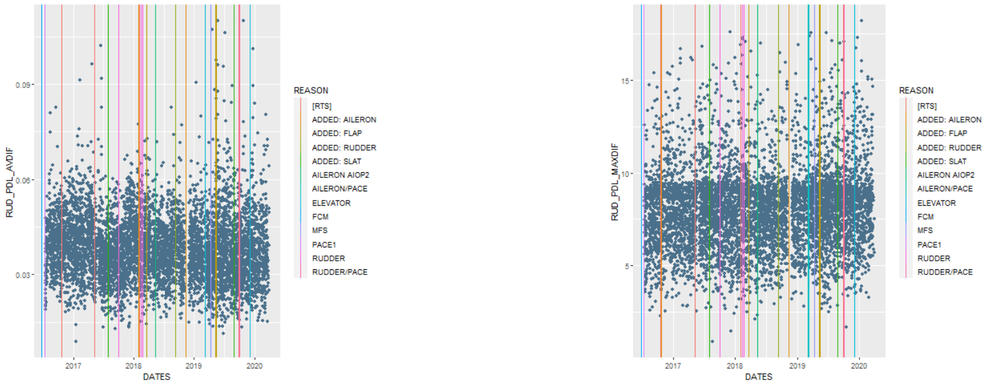
(a) (b) (c)

Figure B.9: Plots for the ELEV_MINFC (left), ELEV_AVFC (centre), and ELEV_MAXFC (right) features on a single aircraft

Finally, regarding the elevator force features, both ELEV_MINFC (Fig. B.9a) and ELEV_MAXFC (Fig. B.9c) seem to carry no meaningful information regarding faults or system degradation, despite there being some behaviour changes. For instance, in Fig. B.9c, there are clear differing distributions: one starting at the last quarter of 2017 and ending around the first quarter of 2018, and the other being the rest of the scatter plot — no reason was found for any of these behaviour changes in the maintenance reports (possibly the FCM3 fault marked the end of the abnormal behaviour, but it is not certain), and this happens in all other aircraft. In fact, ELEV_MAXFC is for the majority of aircraft random and these behaviour changes are very rare, yet when they exist, no source was found.
As for ELEV_AVFC, Fig. B.9b shows not only the previously mentioned fault correlation to the FCM3, but also an increasing trend throughout 2018 ending in December of 2018 where some elevator components were found with physical damage, and were hence replaced (in the plot, this fault is marked as an Aileron fault, however there was also this one in the same day).

In conclusion, only ELEV_AVFC was kept, as it showed in some aircraft both degradation and correlation to faults on the Elevator and Electrical Systems.

B.4 Rudder

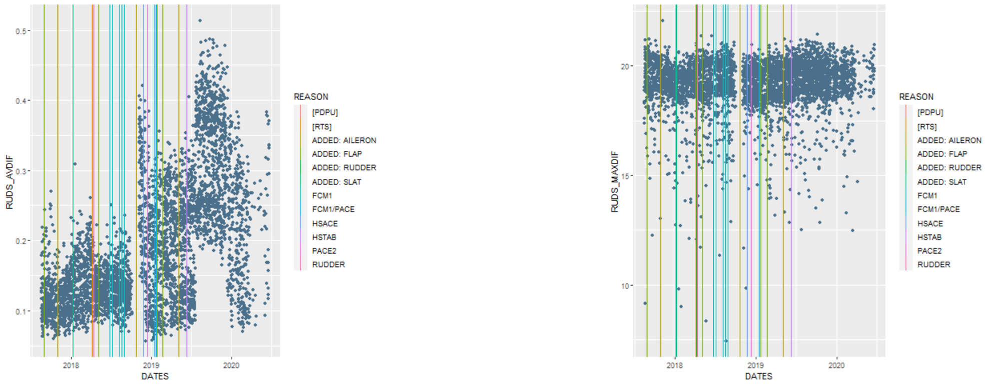


(a)

(b)

Figure B.10: Plots for the RUD_PDL_AVDIF (left), and RUD_PDL_MAXDIF (right) features on a single aircraft

Concerning the features pertaining to the difference between the two pedal readings (see Fig. B.10), all of them were discarded. Both RUD_PDL_AVDIF and RUD_PDL_MAXDIF are mostly random, and RUD_PDL_MINDIF is a constant of zero.

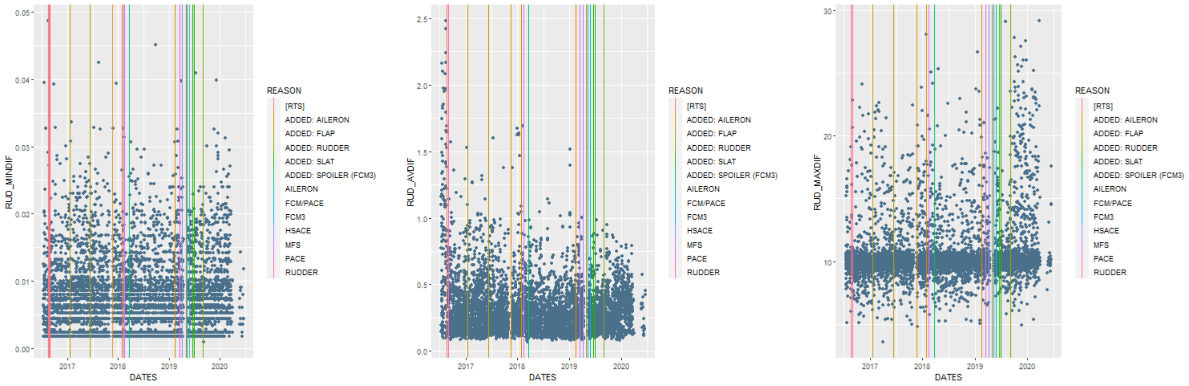


(a)

(b)

Figure B.11: Plots for the RUDS_AVDIF (left), and RUDS_MAXDIF (right) features on a single aircraft

As for the rudder surface sensors reading disparity features (Fig. B.11), RUDS_AVDIF was kept for its most evident apparent correlation to rudder faults. For instance, in Fig. B.11a, an increasing trend from the last quarter of 2017 and extending towards the middle of 2018, seems to have led to, or at least was corrected after, successive faults on the rudder, followed by a rudder PCU replacement. Another example on the same plot would be the behaviour acquired after the maintenance check on the last quarter of 2018, comprised of relatively high values on the feature, closely followed by a fault of a P-ACE. What seems to compromise the usefulness of this feature, however, is the fact that not long after, it assumes values quite higher than those before the aforementioned faults, yet no other issues were reported. Still, it was kept. As for RUDS_MAXDIF, it seemed to bear no more information than its average counterpart; RUDS_MINDIF was always zero.



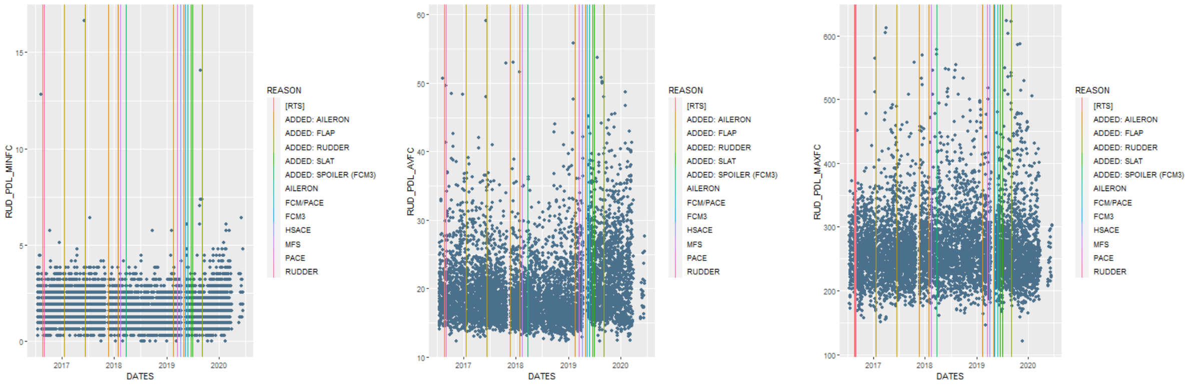
(a)

(b)

(c)

Figure B.12: Plots for the RUD_MINDIF (left), RUD_AVDIF (centre), and RUD_MAXDIF (right) features on a single aircraft

Regarding the rudder features accounting for its input/output relationship (see Fig. B.12), only RUD_AVDIF was kept, as it displayed the greatest apparent correlation to rudder faults, the most flagrant example in the datset being represented in Fig. B.12b, where the first faults in the studied period (rudder faults) seem to stem from the high values of this feature.



(a)

(b)

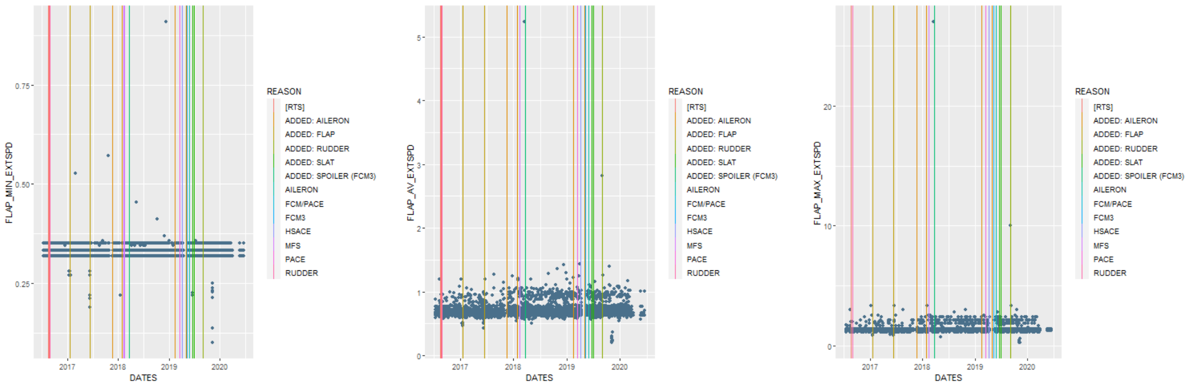
(c)

Figure B.13: Plots for the RUD_PDL_MINFC (left), RUD_PDL_AVFC (centre), and RUD_PDL_MAXFC (right) features on a single aircraft

Finally, none of the rudder pedal force features was kept, as none of them were found to strongly relate to any rudder faults, neither was there any complaint on the rudder pedal stiffness to note on the maintenance reports.

B.5 Flaps/Slats

The flap and slat subsystems are here discussed together as their respective feature analyses are the same regarding fault correlation (flap feature plot characteristics leading to flap faults are the same as slat feature plot characteristics leading to slat faults, and in fact flap feature plots are generally quite similar to their slat equivalents).

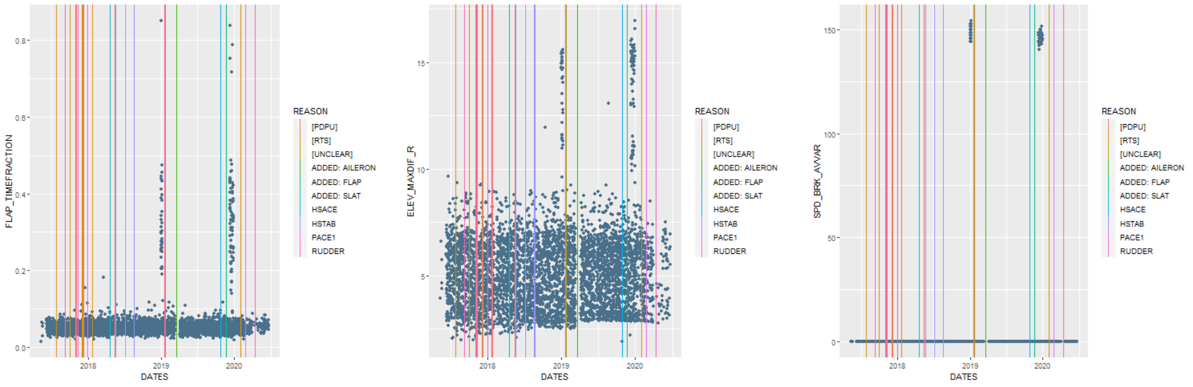


(a) (b) (c)

Figure B.14: Plots for the FLAP_MIN_EXTSPD (left), FLAP_AV_EXTSPD (centre), and FLAP_MAX_EXTSPD (right) features on a single aircraft

Of the features regarding the speed of the flaps to arrive at their correct deflection (Fig. B.14), it was found that the retract and extend feature plots were very similar to each other in shape, differing only in scale. Of all of them, FLAP_MIN_EXTSPD was chosen to represent this system characteristic for two major reasons. On one side, for its clarity in fault correlation, visible in the two flap faults in 2017 in Fig. B.14a with their respective outliers in the plot, indicating the speeds were found to be lower than normal, which is consistent with the fact that most of the system messages found on reports for these kinds of fault mentioned “surface jammed” or “underspeed”. On the other side, the number of possible normal values for this feature was remarkably lower than for the other two features (the apparent three horizontal lines that are formed by the plot points versus the more evident scatter on the other plots) — this would make this feature’s healthy state easier for the autoencoder to learn.
Analogously, the chosen feature for the slats was SLAT_MIN_EXTSPD.

As mentioned before, these speed features were found to be prone to missing values, and initially this was thought to be due to very short flights — such as rejected take-offs, — where the flaps and slats are barely used, if at all. However this was found not to be the case, which prompted individual inspection of the flights which were causing the issue. These flight raw data files, along with the discrete nature of the flap sensor readings, provided a glance at the nature of the aforementioned system-wide peaks.
Indeed, it was found that for these troublesome flights, the sensors were reading both flaps and slats to move irrespective of the flap/slat lever position: the lever could show to stay in position, yet the flap and slat surfaces would move, and it was not possible to calculate a speed of arrival at the theoretical deflection, because there was no means to compute that deflection without lever movement — hence the missing value. So when circumventing this issue by creating the FLAP_TIMEFRACTION and SLAT_TIMEFRACTION features, the following was found, as shown in Fig. B.15:
The problematic flights belonged to the outliers which represent the system-wide peaks. This figure



(a) (b) (c)

Figure B.15: Plots for the FLAP_TIMEFRACTION (left), ELEV_MAXDIF_R (centre), and SPD_BRK_AVVAR (right) features on a single aircraft

shows plots representing features for three distinct subsystems, all having this anomaly, and in truth it affected all of the subsystems.

This discovery helped shedding light on the credibility of these data points: if something as critical as the control surfaces moving on their own were happening, across multiple systems, no aircraft would be flying in any way, shape, or form. In fact, it would have been something the pilots would be aware of, from the very beginning of the flight — in the taxi-out phase, where the controls are tested and the surfaces are checked for movement.

Along with the fact that no reports show any such critical complaints around these periods, and in addition the fact that this seemed to happen at around the same period for all aircraft (see Fig. B.16), this lead to the conclusion that the issue was not in any of the studied systems, and should instead be either a recording, or a software concern.

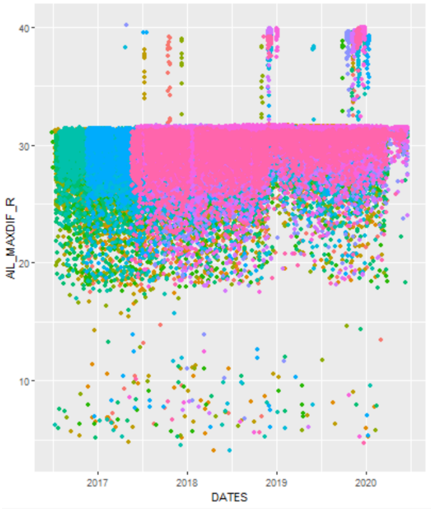
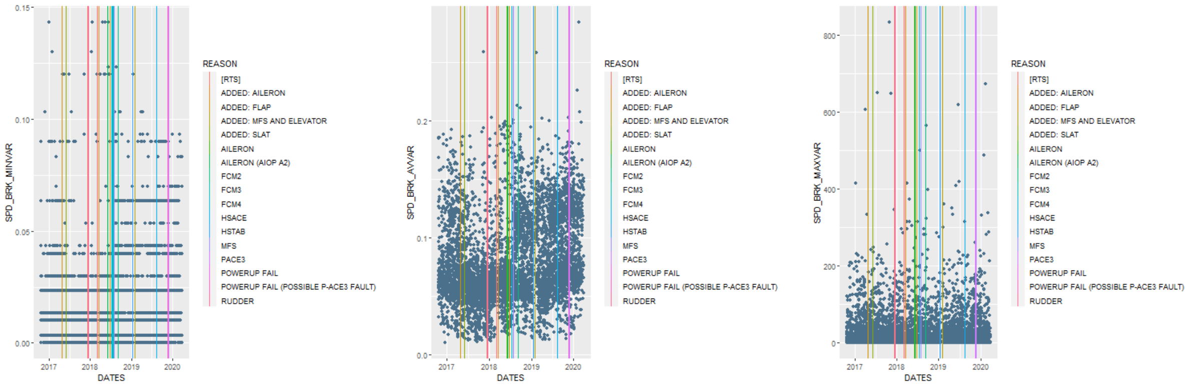


Figure B.16: Plot for the AIL_MAXDIF_R feature for all aircraft, each color representing one aircraft; most system-wide peaks appear at around the same periods

Due to this, it was concluded that system-wide peaks are not representative of the system's health, and that faults happening seemingly connected to them was largely coincidental, as supported by the fact that the vast majority of peaks either is not preceding any fault, or is preceding a fault that is not remotely as critical as this event would be, were it a true fault (for instance, Fig. B.15's peak at the beginning of 2019 seemingly relating to a rudder fault, yet it is known that this fault was exclusive to the rudder).

This is the reason features exhibiting this characteristic were avoided whenever possible, as it was understood the peaks were polluting the data. The same reasoning lead to the discarding of the FLAP_TIMEFRACTION and SLAT_TIMEFRACTION features, with the missing values on the flap and slat speed features being replaced with the mean of the respective features.

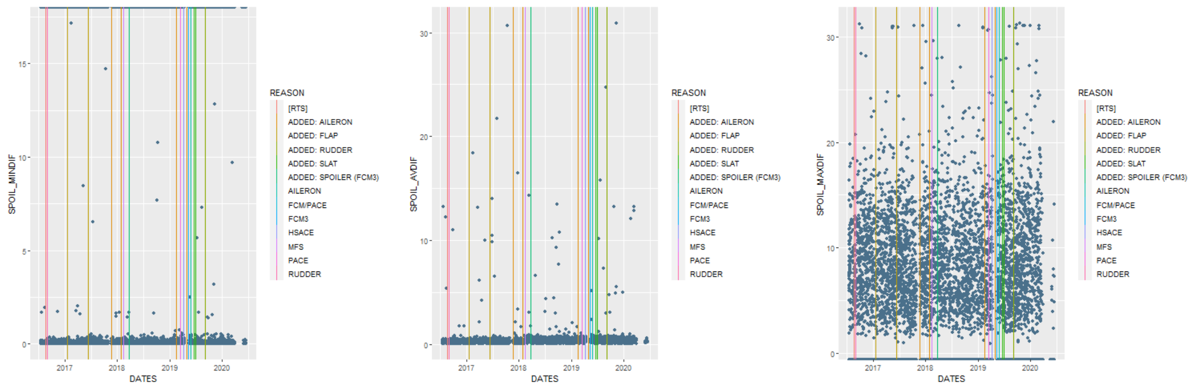
B.6 Spoilers and Air Brakes



(a) (b) (c)

Figure B.17: Plots for the SPD_BRK_MINVAR (left), SPD_BRK_AVVAR (centre), and SPD_BRK_MAXVAR (right) features on a single aircraft

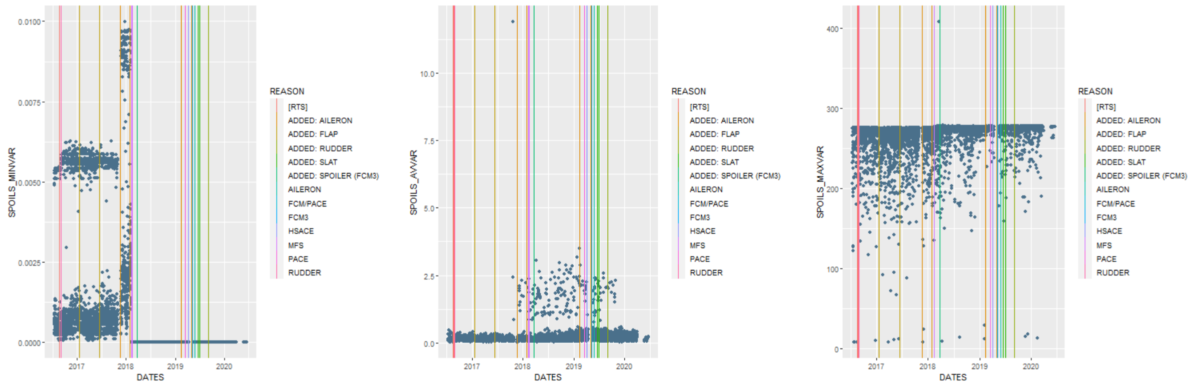
The three SPD_BRK features are illustrated in Fig. B.17, and of them SPD_BRK_MINVAR most evidently showed correlation to multifunction spoiler faults: as exemplified in Fig. B.17a, the first half of 2018 displays a high frequency of higher values than usual, followed by several FCM faults, and finally a multifunction spoiler fault (marked "MFS" in the plot legend). Moreover, SPD_BRK_AVVAR might have also been a choice, however it displays in some aircraft a single outlier several orders of magnitude greater in value than the rest of the plot — likely connected to the system wide peaks — so it was not used.



(a) (b) (c)

Figure B.18: Plots for the SPOIL_MINDIF (left), SPOIL_AVDIF (centre), and SPOIL_MAXDIF (right) features on a single aircraft

SPOIL_AVDIF, SPOIL_MAXDIF and SPOIL_MINDIF (see Fig. B.18) all share the same problem, which is that they rely on pilot input to be able to be calculated. The speed brake subsystem can be either controlled by the pilots, or the auto-pilot, and information about which of these is the agent on the position of the surfaces is given by the AGS variable SPD_BRK, which is 0 when the agent is the auto-pilot, and 1 when it is either of the pilots. The issue arises from the fact that it is not unusual that in a flight the SPD_BRK variable is always 0, meaning there was no instant where the pilots took control of this subsystem. As such, for these cases, a difference between expected and actual surface deflections cannot be calculated, which then translates into missing values on these three features. The frequency of these can be seen at the top of the plot for SPOIL_MINDIF (Fig. B.18a), and at the bottom of the plot for SPOIL_MAXDIF (Fig. B.18c) for they turn into $+\infty$ and $-\infty$ for these features, respectively. As visible, these cases are quite numerous. In fact, almost 50% of all flights carry this characteristic. Given this, and the fact that no obvious visual correlation to spoiler faults was found in any of these features in any aircraft, they were all discarded.



(a) (b) (c)

Figure B.19: Plots for the SPOILS_MINVAR (left), SPOILS_AVVAR (centre), and SPOILS_MAXVAR (right) features on a single aircraft

Of the spoiler sensor reading variance features (Fig. B.19), SPOILS_MINVAR was the one quite notably most representative of fault imminence, for all aircraft. Fig. B.19a shows quite an illustrative example on the matter, where a sudden increase in the values for this feature leads to multiple spoiler faults a few months later, the fault being corrected by the replacement of a PCU. For this reason, it was the only one kept.