Politecnico di Torino

Master's Degree Thesis in Aerospace Engineering

Politecnico
di Torino

Department of Mechanical and Aerospace Engineering

Metaheuristic algorithms for prognostics of on-board electromechanical actuators

Supervisors

Dr. Matteo DALLA VEDOVA

Prof. Paolo MAGGIORE

Eng. Leonardo BALDO

Dr. Gaetano QUATTROCCHI

Candidate

Francesco BATTAGLIA

October 2023

Summary

In recent times the aerospace sector has been going through a deep and significant technological evolution. The main manufacturers are moving towards a new design philosophy: the More Electric Aircraft. The goal is to electrify all the utilities and sub-systems of the aircraft driven by a series of advantages such as, for example, an increase in engine efficiency, a reduction in overall weight, better maintainability, lower operating costs and an alignment with stringent environmental requirements. The development of fully electric flight controls plays a very important role in this design concept. Currently, however, electromechanical actuators are used only in secondary flight controls due to a poor knowledge of failure modes. An unscrupulous use of redundancies would nullify the advantages listed above. The aim of this work will be to identify and study a new prognostic method of these failures, using Bio-inspired metaheuristic optimization algorithms, which can lead us to safety levels at least equal to the current hydraulic system. To do this we will initially simulate in the Matlab-Simulink environment the response of an electromechanical actuator model subject to the imposed failure. Then, with a second simplified model, thanks to the chosen algorithms, we will try to recreate the response generated by the first. The goal will be to minimize the difference between the two, going to solve an optimization problem.

Acknowledgements

Desidero ringraziare la mia famiglia per essere giunto alla fine di questo percorso e tutti i miei relatori per questo lavoro.

Table of Contents

List of Tables

List of Figures

Acronyms

ACU
Actuator Control Unit

BLDC
Brushless Direct Current

CBM
Condition Based Maintenance

DOA
Dandelion Optimizer Algorithm

EHA
Electro-Hydrostatic Actuator

EMA
Electro-Mechanical Actuator

FDI
Failure Detection Identification

FBW
Fly-By-Wire

FCC
Flight Control Computer

FMECA
Failure Mode, Effects, and Criticality Analysis

FTA

Fault Tree Analysis

HBA

Honey Badger Algorithm

LCC

Life Cycle Cost

MEA

More Electric Aircraft

MM

Monitor Model

PDE

Power Drive Electric

PHM

Prognostic Health Management

PID

Proportional Integral Derivative

PWM

Pulse With Modulation

RM

Reference Model

RMSE

Root Mean Square Error

RUL

Remaining Useful Life

SSA

Sparrow Search Algorithm

TLP

Top Level Parameter

Chapter 1

Introduction

1.1 Evolution of flight control systems

Over the years, flight controls have undergone significant advancements. In the early days of aviation, pioneers used wires attached to the flying controls in the cockpit to warp wings and control surfaces for maneuvering biplanes. However, this method was rudimentary and often inadequate. As aircrafts performances improved, articulated flight control surfaces emerged, but the use of wires and pulleys to connect them to the pilot's controls persisted for many years, until it became inadequate for anything beyond basic aircraft (some of them still use this technology)[1].

With the advent of higher speeds (50's)[2], particularly in the transonic region, more complex and sophisticated methods were needed. This was especially true for high-speed fighter aircraft and larger aircraft powered by jet engines. The more the speed increased, the greater the force to be applied to the flight controls became, making the controllability of the aircraft increasingly complex. For instance, the Spitfire experienced high control forces and control reversal issues that were not initially understood[1]. A lack of technological advancement on this front would necessarily have compromised the birth of the modern aircraft that fly the skies today.

To address these challenges, powered surfaces with hydraulically powered actuators were introduced. The following figure 1.1 shows a scheme with the integration of an autopilot system too.

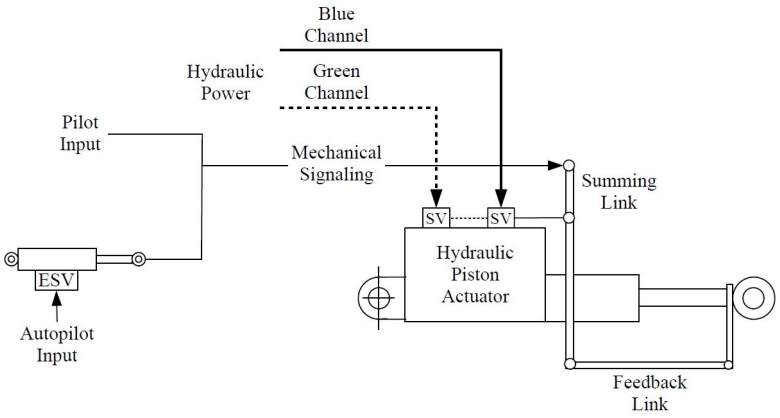

Figure 1.1: Hydraulic actuation system[1]

These actuators enhance the force exerted by the pilot, reducing the physical effort required to control the aircraft and mitigating the effects of higher loads on the flight control surfaces. A further complication of the aircraft required the implementation of control and stability systems (CSAS) made possible only with the help of electronics. The proposed technological advance, combined with the need to reduce weight as much as possible, led to the elimination of a direct mechanical link between the pilot control and the actuator, replaced by an electrical signal: the Fly-By-Wire (FBW) technology (fig. 1.2). The first version of FBW control systems in civil aircraft (80's)[2], which began with the Airbus A320, still had mechanical reversible controls as redundancy (some of them still fly today). In later versions the mechanical reversibility was abandoned. Three examples of aircraft that first adopted this configuration are the Airbus A380, the Boeing 777 and the most recent Airbus A350[1][3].

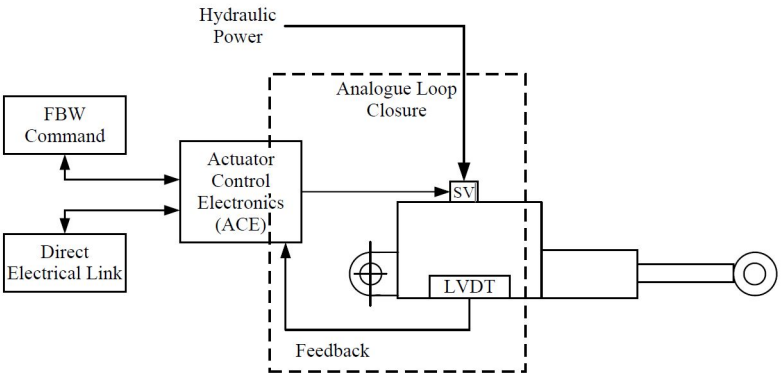

Figure 1.2: FBW actuator system[1]

The most advanced technology currently available is represented by Electro-Hydrostatic Actuators (EHA) (fig. 1.3). Unlike the systems described so far, it aims to reduce energy waste by pressurizing the actuators only when it's necessary. In fact during most of the time of a flight the power required by the flight controls is minimal. This creates a local hydraulic circuit outlined by the hatched area[1]. Unlike figure 1.1 and 1.2 we can see in figure 1.3 the absence of an external hydraulic mandate.

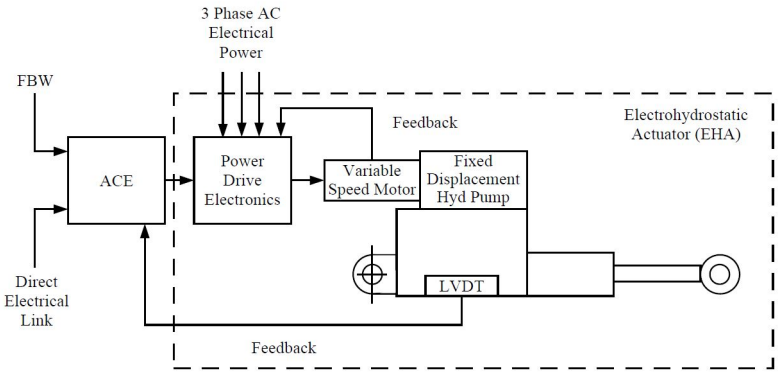

Figure 1.3: EHA system[1]

We can understand, by comparing figure 1.1, figure 1.2 and figure 1.3 the trend to reduce as much as possible the dependence on the hydraulic system until it is completely excluded in favor of electric power. This category of aircraft is called More Electric Aircraft (MEA). But the current state of the art provides that the control surfaces are still electrically controlled but hydraulically powered[4].

1.2 More electric aircraft

1.2.1 Reasons to electrify

The idea is not new: already 30 years ago we started talking about it, but only recently some solutions have found application. Currently the aircraft that produces the most electric power on the market is the B787 with 1450 kVA[5]. This is thanks to the technological progress of recent years and the huge investments made.

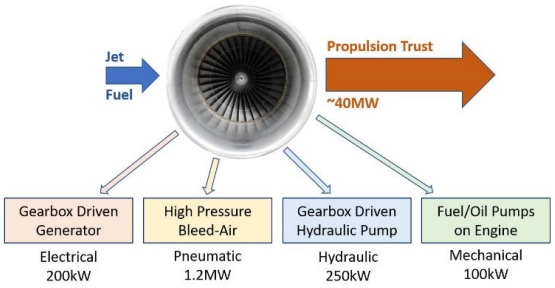

Figure 1.4: Conventional Aircraft[5]

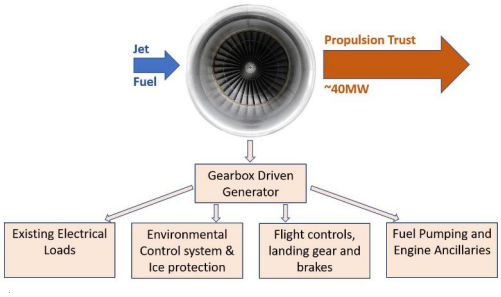

Figure 1.5: More Electric Aircraft[5]

As we can see from the previous figures (1.4 and 1.5) the concept of MEAs extends to all subsystems of an aircraft. In fact, the idea is to electrify all utilities thus reducing the energy taken by the motors beacuse, in conventional aircraft, this energy subtracted from the engine will be balanced with a greater quantity of fuel, to guarantee the same thrust levels. For example, in conventional aircraft, the cabin pressurization system and the de-icing system are powered by a bleed valve that extracts high-pressure air from the engine[5]. In this work we will focus on the flight control actuators: they are the best candidates for replacement given the advantages that their electrification can offer[5][6]. The final challenge for a complete electrification of flight controls will see the replacement of the current EHAs in favor of Electromechanical Actuators (EMA)[7]. But the efficiency of the engine is not the only aspect that drives interest in this change. Hydraulic actuators have a significant drawback in terms of their weight, as they require heavy components such as pumps, reservoirs, and hoses for fluid transportation and motion generation. This additional weight will hold back the development of future aircraft and their performance such as fuel efficiency, range, and payload capacity. In contrast, EMAs are generally lighter and more compact, as they do not rely on hydraulic fluid or associated components, making them an attractive option for weight-sensitive applications[7][6][4][3].

Another limitation of hydraulic actuators is complexity and therefore maintenance requirements. Hydraulic systems involve numerous components, including valves, seals, and filters, which require regular inspection, maintenance, and replacement. On the other hand, EMAs have fewer components and typically require less maintenance, resulting in lower operating costs and reduced downtime for aircraft[7][6][4][3].

Furthermore, EMAs offer improved precision and control compared to hydraulic actuators. They provide finer and more accurate control over motion and torque, allowing for precise and responsive flight control inputs. This level of precision is particularly critical in modern aircraft that rely on advanced flight control

systems, such as FBW technology, which replaces mechanical linkages with electronic controls. EMAs can seamlessly integrate with these advanced systems, enabling more sophisticated control algorithms and enhancing aircraft safety and performance[7][6][4][3].

In addition to these advantages, EMAs also align with the increasing focus on sustainability and environmental concerns in the aviation industry. The potential for fluid leakage and environmental impact associated with hydraulic actuators can be mitigated by adopting EMAs, which do not require hydraulic fluids. This can contribute to reducing the environmental footprint of aircraft and aligning with global efforts to promote sustainable aviation[7][6][4][3].

1.2.2 Limitations and proposed solution

Despite the many benefits and simplifications that can be found in adopting a MEA philosophy, especially with the implementation of EMAs as an actuation system for flight controls, their application is nowadays reserved exclusively for secondary flight controls such as, for example, high-lift devices or air-brakers (i.e. non-safety-critical applications)[4][8][9]. The problems encountered by implementing an EMA are completely different when compared to hydraulic actuation, both from a mechanical point of view and from an electronic point of view: the EMAs are sensitive to electromagnetic disturbances (EMC), they may experience mechanical jamming and overheating issues, which can be attributed to the high currents[3]. Among the many, jamming is one of the most frequent and dangerous, as we will see later.[4]. In fact, an undesired locking of the mobile surface in a given position would inevitably lead to the total uncontrollability of the aircraft. In general, failures in electrical systems, unlike in a hydraulic system (e.g. loss of fluids), do not give any sign before they happen. Furthermore, being a relatively new technology, the lack of data does not help in the study of failures[9].

To ensure a level of reliability at least equal to hydraulic actuation systems, one could think of adding redundancies[7]. However, it is immediate to understand that this solution would lead to an overall complication of the system, nullifying the advantages set out in the previous paragraph.

To pursue the required levels of safety and reliability is necessary to follow another path: a new methodology of failure management called Prognostic Health Management (PHM) is coming our way. It is fundamentally based on the extraction of data and information directly from the system under examination through the numerous sensors distributed in it. The purpose of the PHM is first of all to evaluate the state of health of the system, that is to detect and identify the failures. This phase is called failure detection and identification (FDI). The technique must also anticipate the occurrence of a potentially catastrophic failure in such a way

as to be able to estimate a remaining useful life (RUL) of the component. Should this failure management concept prove useful for achieving safety standards, a large-scale implementation of EMAs as primary flight command would finally be possible. Not only that, new maintenance strategies would be considered going to improve mission readiness, RAMS capabilities and an overall contraction of life cycle cost (LCC)[7][10][11][6][12][13][14][15].

In this work we intend to implement the FDI by solving an optimization problem. Two models will be implemented in Matlab-Simulink: the first, the Reference Model (RM), has the objective of simulating the response that would be presented by the real actuator subject to a failure. The second, the Monitor Model (MM), will be iterated several times by the metaheuristic algorithm until its answer faithfully replicates that of the RM. Once the convergence between the two has been obtained, the parameters of the monitor model will be analyzed to evaluate the state of health of the actuator.

After a general overview of the topic given in this chapter the work will be organized as follows: in chapter 2 we will describe the PHM philosophy; in chapter 3 we will introduce the models that we will use for our prognostic work; in chapters 4 and 5 we will introduce and apply metaheuristic algorithms respectively; in chapter 6 we will comment on the results proposed in the previous chapter and what the future developments could be.

1.3 Overview of EMA

Before continuing with the work and introducing the concepts of PHM it is necessary to go into the detail of an EMA. In this section we will explain and analyze the components of an EMA and how they interact with each other.

We can distinguish two types of EMAs: linear and rotating[16]. For the former, a mechanical component (the Screw in figure 1.6) will be needed to transform the rotary motion into linear motion. In each version of the actuator the electric power will be transformed into mechanical energy. The next figure 1.6 gives us an idea of the classification.

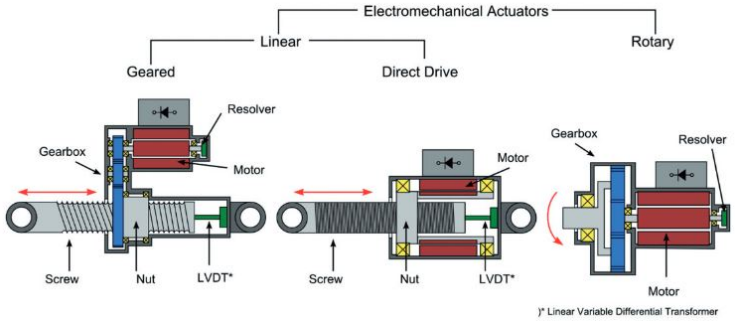

Figure 1.6: EMA classification[16]

Please refer to [16] for further details on subdivisions and applications. In this section, as mentioned before, we illustrate the main components of an EMA by leaning on the figure 1.7. This study will be useful to us in chapter 2 when we discuss EMA's failures.

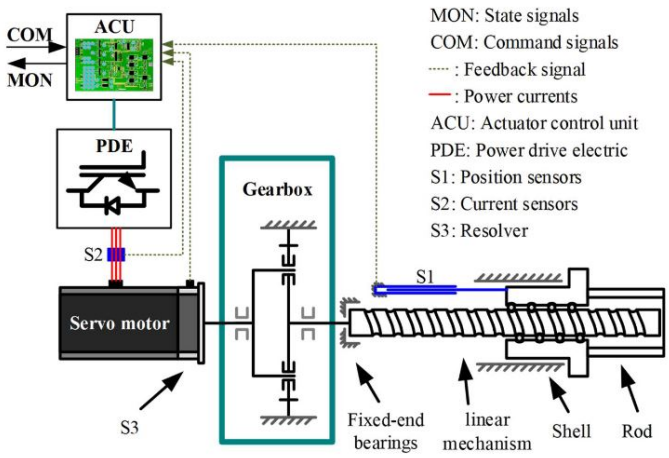

Figure 1.7: EMA scheme[14]

The electrical signal coming from a pilot's command or from the Flight Control Computer (FCC) will meet in sequence the following components [14]:

- *ACTUATOR CONTROL UNIT (ACU)*:
  The ACU works powered by an external electrical power at 28 Volts[16]. Its task is to take the command signal from the flight control computer as input and generate a pulse with modulation signal (PWM). Usually the proportional integral derivative (PID) control algorithm is integrated into the ACU. The PID works using the three-loop of feedback (position, speed and current) that we can see indicated with the dotted line in figure 1.7. The ACU can be

equipped with a diagnostic and prognostic system that, thanks to several sensors distributed in the actuator, can send the health status of the EMA to the FCC.

- *POWER DRIVE ELECTRIC (PDE)*:
  The PDE generates a three-phase current that feeds directly to the motor (red lines in the figure) by taking the input from the ACU.

- *ELECTRIC MOTOR*:
  Main component of the eletromechanical actuator capable of transforming electrical energy into mechanical energy. Among the most common choices are permanent magnet synchronous motor, brushless DC motor and switched reluctance motor[16]. But brushless dc (BLDC) motor are the most used because of their reliability and performance[14].

- *GEARBOX AND SCREW MECHANISM*:
  The first step is to reduce the motor output speed and increase the torque. Task reserved for the gearbox which can be of various types such as harmonic gear reducers, cycloidal reducers, or planetary gear reducers[16]. Two types of mechanisms are generally used to convert rotary motion into linear motion: ball screw mechanism or a planetary roller screw mechanism[16].

- *SENSORS*:
  As mentioned earlier in the EMA several sensors can be deployed with multiple purposes: feedback and health monitoring. All of them will report to the ACU the data that will use them to correct the command given and inform the FCC of the state of health of the system. The parameters to be taken into account for PHM purposes are: position and angle signals, current and voltage signals, temperature signal, vibration signal and torque signal[14].

Chapter 2

Prognostic and health management

2.1 Failure management concepts

As mentioned in the previous chapter, the development of an effective failure management and prevention system seems to be the most promising way to allow a complete and safe electrification of the flight controls, without having to resort to adding redundancies. Even before this need, the process of managing failures and therefore of the related maintenance strategy has undergone a profound evolution over the years not only in the aerospace field, but in general in the industrial field[17] as we can see in the figure 2.1.

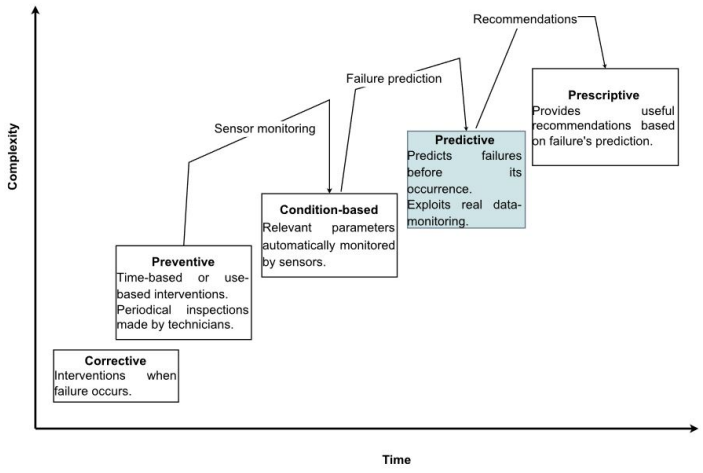

Figure 2.1: Evolution of maintenance strategies over the years[18]

The incessant increase in the complexity of industrial processes and in our case of aerospace products, has highlighted how much the operating costs of a system can have a decisive influence on the Life Cycle Cost (LCC)[18][17]. Only maintenance in the life cycle of an aircraft accounts for 12% of all LCC [19]. The need for a more efficient maintenance methodology has allowed the development of more proactive philosophies such as for example condition based maintenance (CBM) and PHM[18], bringing advantages like better decision making during operations, increased efficiency, and cost savings resulting from reduced ground maintenance and repair work, as well as fewer unexpected faults[17][13]. The emphasis of PHM is on employing analytical methods based on historical data and measurements made in real time that can detect, isolate the fault and predict the state of health of the system so effectively predict the RUL[17][13]. The goal is to obtain a PHM system that can generate the safety conditions such that the adoption of the EMA as an actuation system for primary flight controls is possible[12][6][15].

2.2 Fundamental of PHM

2.2.1 How PHM philosophy works

In accordance with the definition given by [12] and [17], PHM is divided into several macro-phases to reach the desired output which we can initially summarize in figure 2.2.

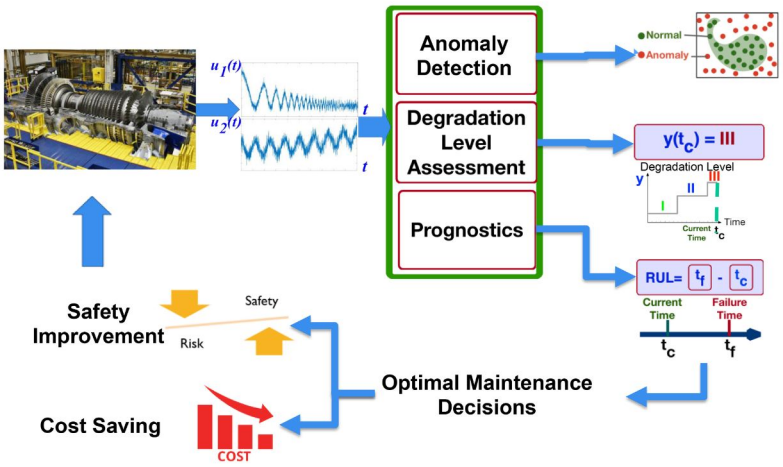

Figure 2.2: PHM tasks[10]

It's easy to guess a greater complexity in implementing such a philosophy than a maintenance scheduled in advance such as preventive maintenance. Furthermore, this approach is often applied to extremely complex systems. This is due both to

the high number of interconnected subsystems, whose degradation processes are not fully known yet, and to the high non-linearity of the response [10]. It will therefore be essential to design the most appropriate PHM strategy (there are many different formulations) already from the early stages of conceptual design of the system[10].

Referring to figure 2.2 and keeping in mind what was discussed in [17] e [10], in the next sub-paragraphs we go into the details of the phases of a PHM approach.

2.2.2 Data acquisition

The first task of the PHM system is to collect information such as to be able to subsequently carry out the real diagnostic and prognostic work. This acquisition usually takes place through real-time measurements of parameters that we are interested in monitoring, such as pressure, temperature, vibration, etc. of a component. We therefore reiterate the importance of creating a PHM system suitable for the needs required right from the early stages of conceptual development of the aircraft. The measurements made can be classified into: time-domain, frequency-domain, and time-frequency-based. The former is suitable for better performing the fault detection task[17]. Unfortunately, the process of acquiring the necessary data is not without problems. In fact, it is necessary to take into account measurement errors by malfunctioning or badly calibrated sensors, incompleteness and lack of measurements, the difficulty of managing large amounts of data (big data), heterogeneity among them and finally the different operating conditions under which the system can be submitted. In addition to the classic integrated sensors, more advanced wireless types can also be installed. In any case, all refer to a central unit which has the task of collecting the measurements[13]. Before moving on to the next step, it is necessary to carry out a pre-processing of the information collected and to extract its characteristics. The data will be filtered to eliminate any disturbing noise and combined with each other if coming from different sources.

2.2.3 Analyse

We are now referring to the green box in figure 2.2. Once the necessary information has been acquired, the real work of PHM takes place. To carry it out, various techniques have evolved over time and others, that are even more efficien, are the subject of research and study today. Take for example the adoption of artificial intelligence and machine learning.

The analysis phase includes the three key steps which lead us to the implementation of condition-based and predictive maintenance. They are:

- FAULT DETECTION: Fault detection is the operation through which the presence of anomalies in a component is evaluated with respect to normal

operating conditions (the nominal condition). The process consists of a continuous monitoring cycle and if the difference between nominal condition and the state of degradation measured exceeds a certain threshold, the component is declared *failed*[20].

- FAULT IDENTIFICATION: Once the presence of a fault has been ascertained, it is necessary to establish the level of degradation, the location of the failure and what are the causes that caused it[10].

Fault detection and identification can be grouped under the name of FAULT DIAGNOSTIC[13].

- FAULT PROGNOSTIC: The most challenging operation is the prognostic of failures. In this phase we try to predict the evolution over time of the state of degradation of the component and therefore to estimate its RUL. Although some types of failures that occur intermittently and randomly make this phase delicate and more complex than fault detection[18][17], the goal is to achieve a certain level of confidence in predicting failures to fully exploit the benefits that the PHM philosophy can bring.

2.2.4 Maintenance decision

Once the PHM strategy that is deemed most appropriate has been implemented, through the results and information that it provides us regarding the state of health of the system, it is possible to plan more personalized maintenance such as CBM and predictive maintenance[3][17][21]. Among numerous benefits already discussed in the previous section, the one taken into consideration in this work is the increase of safety conditions. Necessary requirment to extend the EMAs to the primary flight controls.

2.3 PHM for EMAs

Now having a clearer vision of what the PHM philosophy is, in this section we want to proceed with studying its application for the management of maintenance and failures of an EMA. Let's start by evaluating the general implementation procedure with the scheme in figure 2.3, bearing in mind what was also said in the previous section.

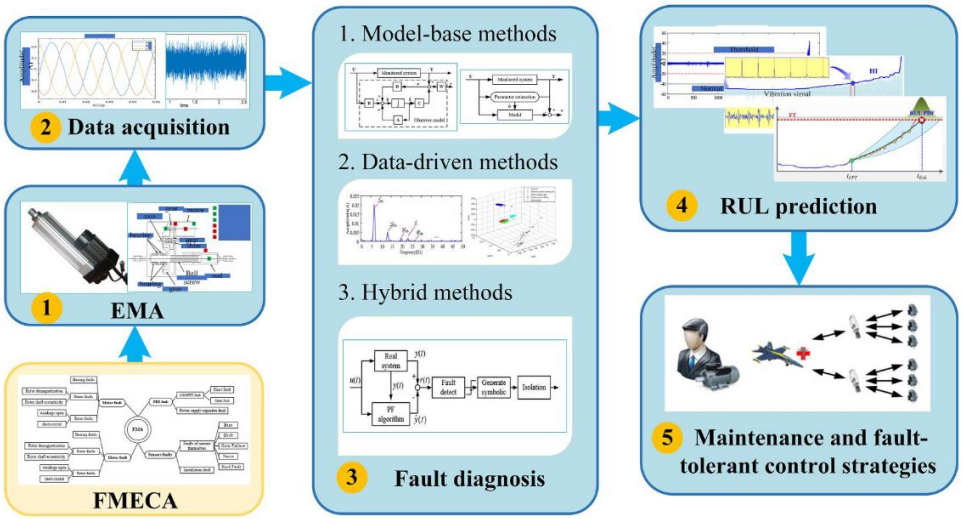

Figure 2.3: General procedure for EMA [14]

As we know, the prognostics and diagnostics of failures require a correct collection and extrapolation of information through the sensors distributed within the EMA. But to design an efficient PHM strategy it becomes of fundamental importance to classify and prioritize the failures that may occur[6]. This task is usually performed by following two proposed failure analysis methodologies: the Failure Mode, Effects, and Criticality Analysis (FMECA) and the Fault Tree Analysis (FTA). In paragraph 2.3.1 we will deal with reporting the classification of failures of an EMA while in paragraph 2.3.2 we will discuss the different approaches to carry out the PHM task.

2.3.1 EMA failure analysis

We have said that unlike the hydraulic system, the electromechanical actuator presents a completely different series of problems related to the management of failures. The absence of any premonitory signals for some of them is one of the reasons that limit the application of the EMAs to secondary flight controls. This is why, before developing a diagnostic and prognostic system to overcome these drawbacks, it is necessary to have an in-depth knowledge of all the failures that may occur. The FMECA carried out by [11] highlights 1950 failure modes that can initially be grouped into four categories as proposed by [14] and [22]. They are reported below in figures 2.4, 2.5, 2.6, 2.7.

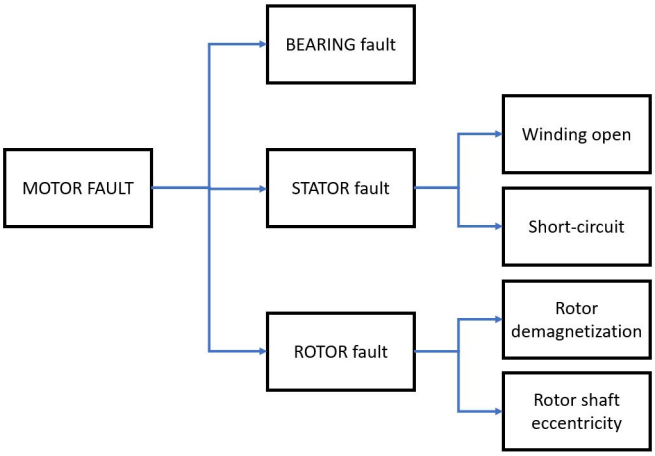

Figure 2.4: Motor fault[14]

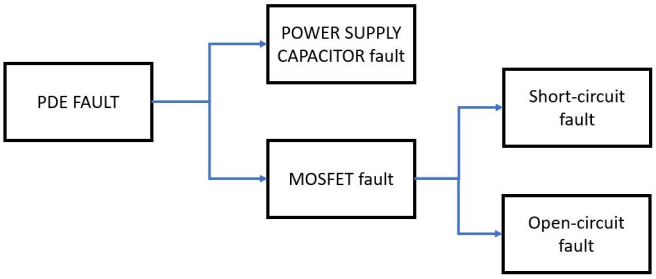

Figure 2.5: PDE fault[14]

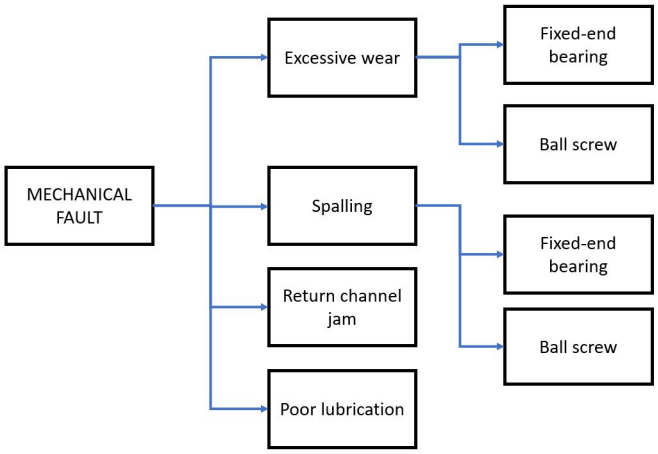

Figure 2.6: Mechaniacal fault[14]

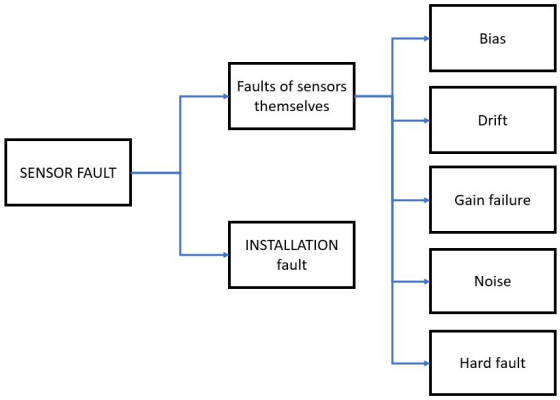

Figure 2.7: Sensors fault[14]

- MOTOR FAULT: The main causes of failure can be bearing fault, stator winding fault, and rotor fault. The first is the one that occurs most frequently (about 41% of total). If the eccentricity level exceeds a certain value, we may have contact between the rotor and stator as well as imbalances and strong vibrations. The high rotation speeds generate heat which, if not properly dissipated, leads to significant mechanical stress which can in turn cause eccentricity. In addition, heat can lead to short-circuiting of the windings [14][22].

- PDE FAULT: The control electronics present types of failure substantially identical to the other electronic parts on board the aircraft. The most sensitive components to possible failures are the H-bridge MOSFET and power supply capacitor. Also in this case the thermal cycles can create problems such as short circuit and open circuit which respectively produce an excessive increase in current (catastrophic event) and a phase loss with consequent loss of torque. It is logical to understand how a loss of the control electronics can lead to the ungovernability of the actuator[14][22].

- MECHANICAL AND STRUCTURAL FAULT: The two components that are most prone to failure are the ball screw pair and fixed-end bearings. Mechanical failure mainly caused by excessive load, environmental factor, poor lubrication and manufacturing defect. An example of a potentially catastrophic failure is the seizure of the ball screw pair often due to one or more of the causes listed above[14][22].

- SENSOR FAULT: Sensors are an integral part of the control loops. They are therefore sensitive to installation errors that can degrade control performance. In addition to the installation errors, we list below the five types of failures with the relative consequence in brackets: bias (output has a constant offset),

drift (the offset of the output is time-varying), gain failure (the gain multiple changes with time), noise (the output is a random sequence) and hard fault (the output is fixed to a constant value). The main cause of these failures is the hostile environment in which the sensors work[14][22].

The FMECA carried out by [22], offers us a list of all the possible failures that can occur in an EMA. The goal is therefore to find out what the triggers are for each of them. Task entrusted to the FTA which will take as input the single failures proposed by the FMECA[11]. The FTA is a logical diagram that seeks to find all the causes of a single failure, how they are connected and what probability they have of occurring. The study carried out by [11] identifies four fundamental safety requirements for an EMA, shown in table 2.1. Furthermore, the failure rate expressed in $\frac{1}{FH}$ and the requirement imposed, always expressed in $\frac{1}{FH}$, are reported for each. Where FH stands for flight hours.

Table 2.1: Requirements[11]

TOP FTA EVENT	FAILURE RATE	REQUIREMENT
Actuator loss of control/function	$6.218 \cdot 10^{-6}$	$< 10^{-7}$
Actuator free floating	$6 \cdot 10^{-9}$	$< 10^{-7}$
Actuator runaway	$2.071 \cdot 10^{-12}$	$< 10^{-8}$
Actuator jam	$3.648 \cdot 10^{-8}$	$< 10^{-9}$

We can see that a failure management and monitoring system is necessary since the requirements of *Actuator loss of control/function* and *Actuator jam* are not satisfied.

2.3.2 PHM methods for EMA

The application of a diagnostic and prognostic philosophy can follow different approaches available today (fig. 2.3). We can group them into three different categories: data-driven, model-based and hybrid (a fusion of both) approaches[13][14][17]. Choosing one of them will depend on the type of system, the type of diagnostics to be implemented and the type of data available. In fact, each of them has advantages and disadvantages and we cannot therefore establish that a certain methodology can always be better than another. Let's take a closer look at the main features of the first two. This will make it clear why we will adopt a model-based philosophy for our work.

- DATA-DRIVEN: Preferred approach for systems for which you do not have in-depth knowledge of physics but have a rich set of failure data. These can be historical observations, current condition monitoring data and failure data. The health conditions can be obtained through the elaboration and

classification of the feature extracted from these data and measurements made. The data-driven technique has the following strengths: low costs and speed of algorithm development. The main disadvantages are the difficulty in processing large amounts of data, the difficulty in obtaining them (for example in the aerospace field) and the possible inaccuracies that may arise from the processing[13][14][4].

- MODEL-BASED: The opposite methodology to data-driven is model-based. The physics and dynamic response of the EMA is approximated using a mathematical model. The response provided by the simulation of the model, which should represent the nominal conditions (usually defined by tests carried out previously), is compared to the behavior assumed by the real actuator in operation. The deviation from the nominal conditions will indicate the state of health of the system. This leads us to disengage from the knowledge of a large set of historical data on failures. Furthermore, the accuracy can be easily managed according to the user's requests.
  Compared to the data-driven approach, this technique requires detailed knowledge of the system (in this case of the EMA) to conceive a sufficiently suitable model and that it does not undergo excessive simplifications. A task that is increasingly time consuming as the system becomes more complex[13][14][4].

As mentioned in chapter 1, given the difficulty in finding sufficient data about the failure modes, our choice falls on a model-based approach. The models adopted will be illustrated in the next chapter.

Chapter 3

Electro-mechanical actuator models and fault implementation

3.1 Model-based adopted procedure

We said that thanks to a model-based approach we are able to estimate the health conditions of the actuator through the development of mathematical models. The idea is to carry out the check during the various phases of the aircraft's operational life, from the pre-flight walk around to the cruise phase. The general procedure requires that the dynamic response of a real actuator is compared with that proposed by the monitor model, which parameters will indicate the deviation from the nominal operating conditions[14].

In our work, however, we will implement two mathematical models. The first, due to the lack of sufficient data, will replace the real actuator with a high-fidelity model that we call Reference Model (RM). It will act as a numerical test bench in which the failures will be simulated. A second low-fidelity model that we will call Monitor Model (MM) characterized by top level parameters (TLP) (indicated with the letter k in figure 3.1), each one for each failure to be monitored. The health check is performed by iterating the MM several times, modifying the TLPs at each iteration. The purpose of the iterations is to find those that minimize the difference between RM and MM and the error between the two is defined by a suitable fitness function. Once the minimum is reached, the TLPs (k_{opt}) will be processed to assess the health of the system and evaluate incipient failures.

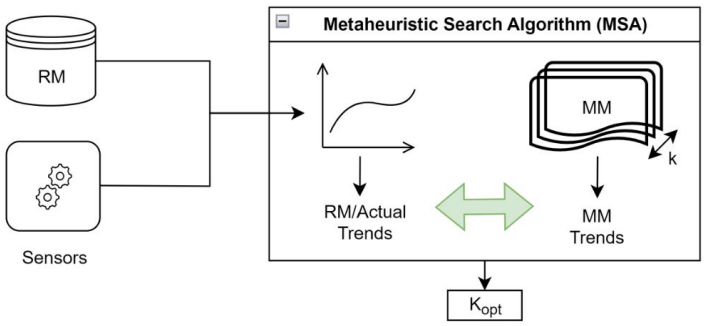

Figure 3.1: Model-based methodology[3]

The search for TLPs that minimize the error is, to all intents and purposes, an optimization problem whose resolution will be entrusted to bio-inspired metaheuristic algorithms. They will be explained in more detail in chapter 4 and applied in chapter 5. Both models will be executed in Matlab-Simulink environment and the running time will be another parameter not to be underestimated (currently the main limitation to perform a diagnostic of this type during the cruise).

In the following sections (3.2 and 3.3) we will fully describe the structure and logic of the two models just mentioned. They are supplied to us by the *Mechanical and aerospace engineering department* (DIMEAS) of Turin Polytechnic.

3.2 Reference Model

3.2.1 High-level reference model

To replace the real test bench, a model that faithfully replicates its answer will be needed. Task entrusted to the RM which we will deal with in this paragraph. Bearing in mind the description of the EMA made in section 1.3, we report in figure 3.2 a summary diagram of the components and how they interact with each other[20]. The Simulink model will be built following this logic.

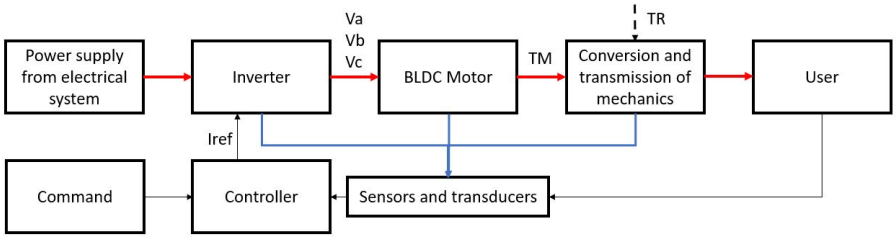

Figure 3.2: Reference Model scheme

As we can see a distinction has been made by colors in the connections of the

blocks: the red arrows represent a high power transmission, the blue arrows a low power transmission and the thin black arrows a signal transmission[20].

The controller, which is of the Proportional Integral Derivative (PID) type, compares the command received as input with the signals coming from the sensors and transducers distributed in the EMA. It will directly control the inverter with the reference current (I_{ref}) which transforms the electrical power supplied by the system into a three-phase current (Va, Vb, Vc) of the Pulse With Modulation (PWM) type. The electric motor will then have to generate a certain speed and torque (TM) based on the PWM signals it receives. Last but not least, the motion conversion and transmission block, taking into account the resistant torque (TR) applied to the aerodynamic surface, provides the user with a desired position command.

In the figure 3.3 we can see the high-level Simulink model and comparing it with the diagram of figure 3.2 we notice an additional block: the *signal acquisition* block. Its function will be explained later.

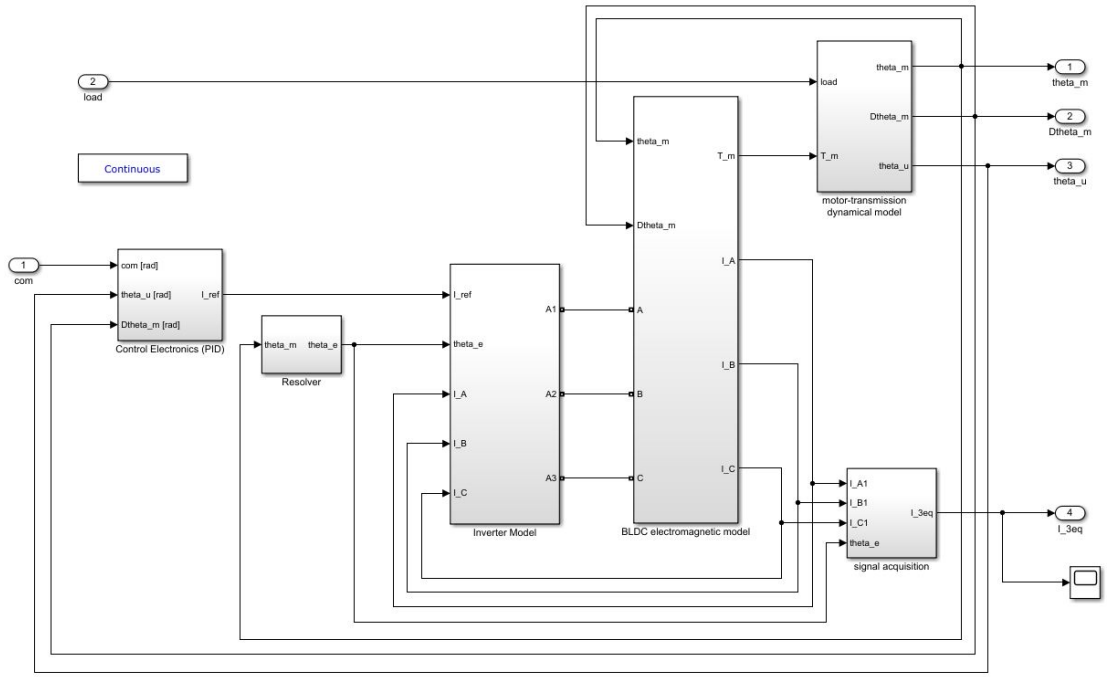

Figure 3.3: High level RM on Simulink

In the following tables we list the parameters and assumed values of the high-fidelity model for future simulations.

Parameter	Symbol	Value	Unit of measure
Error proportional Gain	G_{prop}	$1 \cdot 10^5$	$\frac{1}{s}$
PID proportional gain	K_P	$5 \cdot 10^{-2}$	$\frac{Nms}{rad}$
PID integrative gain	K_I	0	$\frac{Nm}{rad}$
PID derivative gain	K_D	0	$\frac{Nm}{rad}$
Maximum current	I_{max}	22.5	A

Table 3.1: Controller Parameters

Parameter	Symbol	Value	Unit of measure
Maximum supply voltage (DC)	V_{max}	48	V
Hysteresis band width	H_B	0.5	A

Table 3.2: Inverter Parameters

Parameter	Symbol	Value	Unit of measure
Engine saturation torque	Tm, max	1689	Nm
Counter-electromotive force's constant	K_{cemf}	0.0376	$\frac{Nm}{A}$
Nominal phase-to-phase resistance	R_S	2.13	Ω
Phase-to-phase inductance	L_S	$7.2 \cdot 10^{-4}$	H
Number of pole pairs per phase	N_P	2	–
Nominal rotor static eccentricity	ζ	0	–
Nominal rotor static eccentricity phase	ϕ	0	–

Table 3.3: BLDC Motor Parameters

Parameter	Symbol	Value	Unit of measure
Moment of inertia of the motor	J_m	$1.3 \cdot 10^{-5}$	kgm^2
Motor viscous damping coefficient	C_m	$9.549 \cdot 10^{-6}$	$\frac{Nms}{rad}$
User viscous damping coefficient	C_u	$4.507 \cdot 10^{-7}$	$\frac{Nms}{rad}$
Static motor friction	f_{sm}	$0.06 \cdot T_{m,max}$	Nm
Dynamic motor friction	f_{dm}	$0.03 \cdot T_{m,max}$	Nm
Nominal backlash	BLK	$1 \cdot 10^{-5}$	rad

Table 3.4: Transmission Dynamic Parameters

3.2.2 Command and load block

The first two blocks we analyze are the command block and the load block respectively represented in figure 3.4 and 3.5. They are both very similar and allow us to apply different external commands and loads to the actuator. In our work the external load will be set to zero.

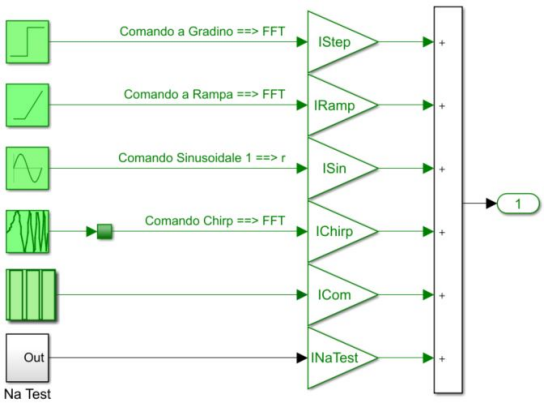

Figure 3.4: Command block

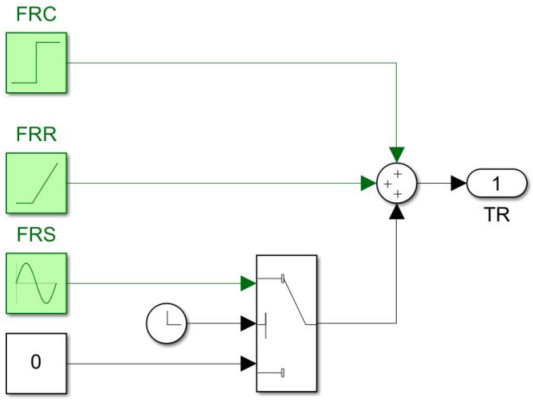

Figure 3.5: Load block

The possibilities range from a simple step command up to the chirp command. You can also implement your own command or provide no command at all. For each command we report the parameters used in the following tables.

Parameter	Value	Unit of measure
Initial amplitude	0	rad
Final amplitude	1	rad
Instant of application	0.01	s

Table 3.5: Step command

Parameter	Value	Unit of measure
Ramp gradient	10^{-3}	$\frac{rad}{s}$
Initial ramp output	0	rad
Instant of application	0	s

Table 3.6: Ramp command

Parameter	Value	Unit of measure
Semi-amplitude input	$5 \cdot 10^{-3}$	rad
Bias input	0	rad
Input frequency	$15 \cdot 2\pi$	$\frac{rad}{s}$

Table 3.7: Sine command

Parameter	Value	Unit of measure
Initial amplitude	$5 \cdot 10^{-3}$	rad
Initial frequency	0	Hz
Target frequency	15	Hz
Instant when the target frequency is reached	0.5	s

Table 3.8: Chirp command

3.2.3 Control electronics block

Let's see in the figure 3.6 the composition of the block on Simulink. A position error is initially calculated ($Err_{pos} = Com - \theta_u$) between the commanded position and the measured position. Subsequently, to obtain the speed error, the position error is multiplied by a proportional gain and saturated at $\pm 8000 rpm$. The obtained value is compared with the real motor speed ($Err_{vel} = \omega_{ref} - \dot{\theta}_m$) and the result will be fed to the PID controller.

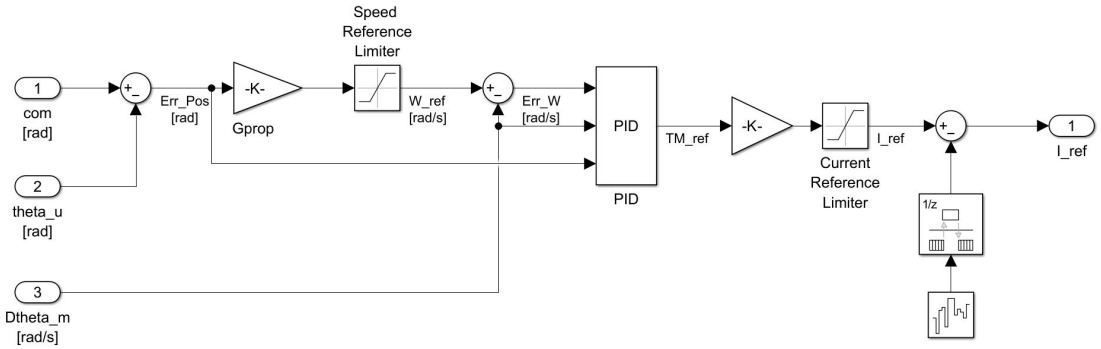

Figure 3.6: Controller Block

Two other parameters are supplied to the PID to play the role of dynamic compensation: the previously calculated position error (Err_{pos}) and the motor speed ($\dot{\theta}_m$). As we know from [23] dynamic compensation via PID occurs through

three actions: proportional to the reference error (defined by the constant K_P), proportional to the integral of the reference error (defined by the constant K_I) and proportional to the derivative of the reference error (defined by the constant K_D) as reported in 3.1.

$$u(t) = K_P e(t) + K_I \int_0^t \epsilon(\tau)d\tau + K_D \frac{de(t)}{dt} = K_P \left(e(t) + \frac{1}{T_I} \int_0^t \epsilon(\tau)d\tau + T_D \frac{de(t)}{dt} \right) \tag{3.1}$$

In the second part of the equation we find a formulation that highlights the characteristic integrative time T_I set to 10000 s and the characteristic derivative time T_D set to 0 s.

To eliminate the wind-up problem, in the PID block, three blocks have been implemented: *velocity* when the speed is zero for at least one integration step, *position* when the position error exceeds a certain established threshold, *saturation* if the requested command reaches the maximum position. For each of these cases the action of the integrative branch is deactivated[20].

At the output of the PID we have the reference torque of the motor, which divided by a proportional gain gives us the reference current (I_{ref}) subsequently saturated at 22.5 A. This current will be the one that will govern the inverter. An additional block of white noise has been inserted to generate random numbers as a source of noise. In our case it will be neglected and its value set equal to zero.

3.2.4 Inverter block

Looking at the high-level diagram in figure 3.3, we can say that the inverter takes as input the currents of the three phases of the motor as feedback, the current I_{ref} generated by the control electronics and the electrical position of the motor θ_e. Thanks to the instructions provided by the I_{ref} signal, the inverter will generate the three phases V_a, V_b, V_c which will directly control the motor.
Before describing the simulink diagram of the inverter, let us pause for a moment on the parameter θ_e. It is calculated by the resolver block. The resolver transforms a mechanical angle (in our case relating to the motor shaft) into an electrical signal. Just like an electric motor, it has a stator and a rotor. Based on the angular position of the latter, the inductive couplings of the stator will energize in a different way, generating an associated electrical signal instant by instant. The mathematical relationship on which the solver is based is the 3.2 that physically links the mechanical angle (θ_m) with the electrical angle (θ_e).

$$\theta_e = 2\pi \left(N_p \frac{\theta_m}{2\pi} - floor \left(N_p \frac{\theta_m}{2\pi} \right) \right) \tag{3.2}$$

Where the Matlab *floor* function rounds each element of a vector to the nearest integer less than or equal to that element [24] and N_p is the number of pole pairs. In Simulink we have the block in figure 3.7.

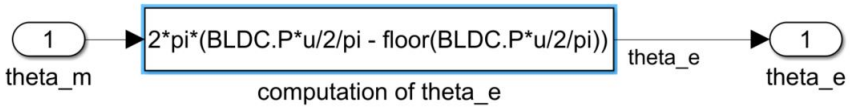

Figure 3.7: Resolver Block

Let's look at the components inside the *Inverter* block with figure 3.8.

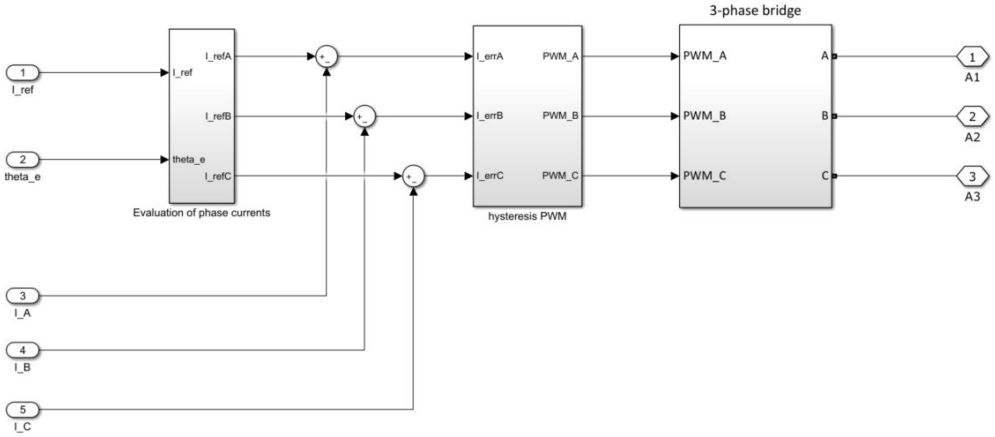

Figure 3.8: Inverter block

We have said that the function of the inverter is to generate the three phases in output to control the electric motor. The I_{ref} and the θ_e just calculated will be taken over by the antitransformate of Park and Clark with the block *evaluation of phase current*, to change the current from rotoric reference to phase current. Before becoming input of the *Hysteresis PWM block*, the three command currents I_{refA}, I_{refB}, I_{refC} must be compared with the three currents taken in feedback from the motor (I_A, I_B, I_C). The *Hysteresis* block will thus be able to generate the three PWM signals which will control the motor. The last necessary step is the three-phase H-bridge that we model in Simulink using the universal bridge.

3.2.5 Motor block

Excluding brush motors, even if practical, the choice for aerospace applications for reliability issues falls on a BLDC motor, which is a type of *synchronous motor*. This means that the magnetic field generated by the stator and the one generated

by the rotor will be synchronized[25]. Among the various versions available, the most widespread and which we will also use is the three-phase one.

As mentioned, the motor consists of a stator, whose task is to house the windings of the three phases and the rotor on which the permanent magnet will be integrated. The two are separated by a thin layer of air to maximize efficiency and save space. However, this leads to careful attention to eccentricity and greater difficulty in dissipating heat. The three phases will be controlled by the output of the inverter just seen (V_a, V_b, V_c). The current flowing inside the windings will generate a magnetic field which will interact with the magnetic field generated by the permanent magnets placed on the rotor. The latter will try to constantly align itself with that generated by the stator. The goal is to always have a 90 degrees angle between the two fields to obtain the maximum torque available. The phases will then be fed in sequence to have a rotating magnetic field of constant magnitude[20].

Now that we have a clearer idea of the physics of a BLDC motor, let's see its implementation on Simulink through figure 3.9.

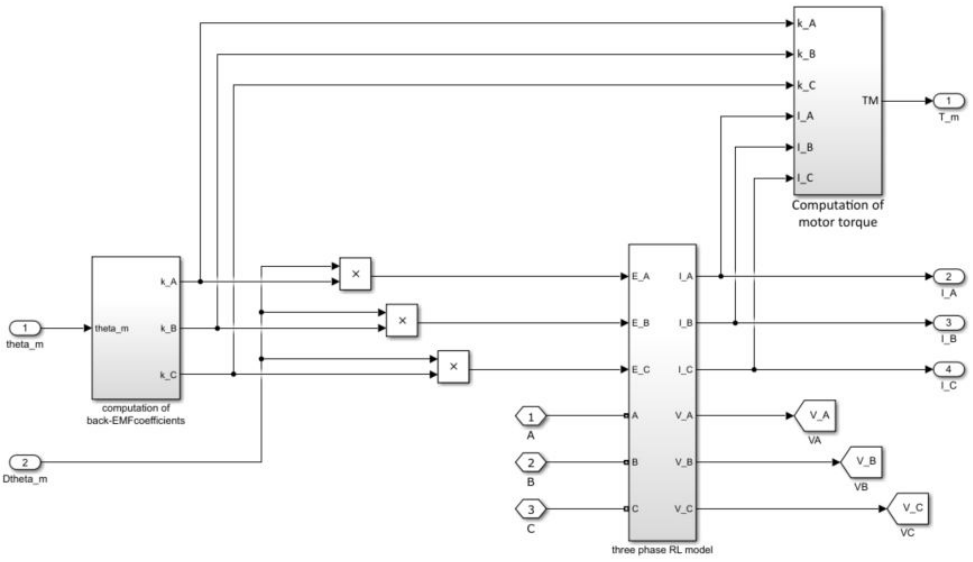

Figure 3.9: Motor block

The three constituent blocks are: *computation of counter electromotive force*, *three phase RL model* and *computation of motor torque*.
The counter electromotive force mainly depends on the position (angle) of the motor θ_m. Since a failure we have discussed concerns the static eccentricity of the rotor, we will express with the following equations (3.3, 3.4, 3.5), the counter electromotive force as a function of it, in terms of modulus ζ and phase ϕ [26].

$$k_{A,cemf} = N_A k_{fcem}(-\sin\theta)(N_p+1)\zeta\cos(\theta_e-\phi) \tag{3.3}$$

$$k_{B,cemf} = N_B k_{fcem}\left(-\sin\left(\theta-\frac{2}{3}\pi\right)\right)(N_p+1)\zeta\cos(\theta_e-\phi+\frac{2}{3}\pi) \tag{3.4}$$

$$k_{C,cemf} = N_C k_{fcem}\left(-\sin\left(\theta-\frac{4}{3}\pi\right)\right)(N_p+1)\zeta\cos(\theta_e-\phi+\frac{4}{3}\pi) \tag{3.5}$$

The coefficient is calculated for each phase and translated by $\frac{2}{3}\pi$ for the second phase and $\frac{4}{3}\pi$ for the third phase. The term N_i, on the other hand, is set to 1 if current flows in the phase, otherwise it is null. The term ζ instead, i.e. the modulus of eccentricity, is calculated with 3.6:

$$\zeta = \frac{x_0}{g_0} \tag{3.6}$$

Where x_0 is the distance between the stator axis and the rotor axis, while g_0 represents the distance under nominal conditions between the rotor and the stator (air gap).

The back electromotive force E is calculated by multiplying the coefficients just obtained with the motor speed ω_m:

$$E = k_{i,cemf}\omega_m \tag{3.7}$$

Now the calculated counter electromotive forces and inverter output voltages are supplied to the RL circuit. Each phase will be connected and controlled by two transistors thus forming three RL circuits arranged in a star. We see a representation of the physics of the circuit in figure 3.10.

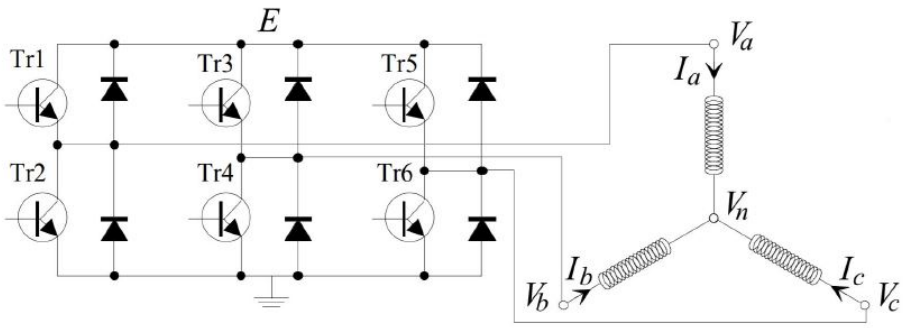

Figure 3.10: RL circuit [26]

The voltage V_i on the generic armature can be calculated using the equation 3.8[20].

$$V_i = L\frac{dI_i}{dt} + RI_i + E_i \tag{3.8}$$

Where L is the inductance of the circuit, I_i is the current flowing in the armature and E_i is the counter electromotive force. Applying the formula to our three-phase motor we will have 3.9.

$$\begin{bmatrix} V_a \\ V_b \\ V_c \end{bmatrix} = L\frac{d}{dt}\begin{bmatrix} I_A \\ I_B \\ I_C \end{bmatrix} + R\begin{bmatrix} I_A \\ I_B \\ I_C \end{bmatrix} + \begin{bmatrix} E_A \\ E_B \\ E_C \end{bmatrix}. \tag{3.9}$$

In figure 3.11 we report the Simulink scheme equivalent to the circuit in figure 3.10.

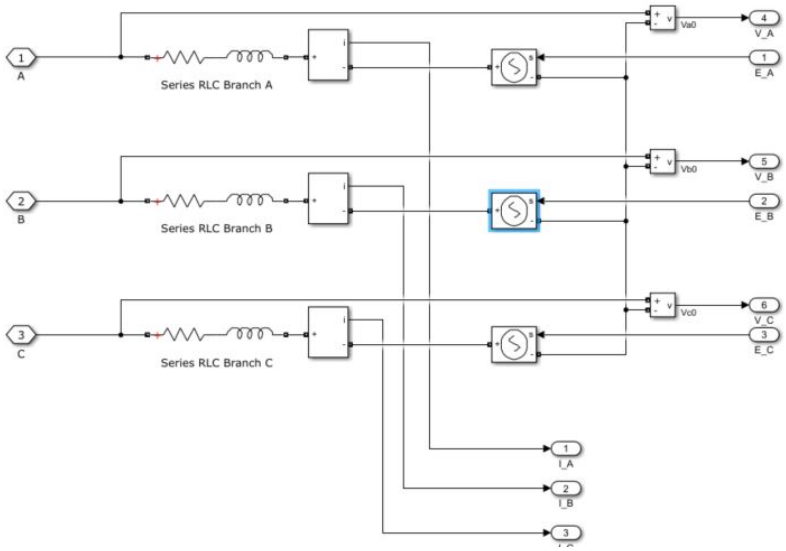

Figure 3.11: RL Simulink circuit

The computation of motor torque final module allows us to calculate the torque output of the motor. The block is constructed following the equation 3.10.

$$T_m = (k_{cfemA}I_A + k_{cfemB}I_B + k_{cfemC}I_C)\sqrt{2} \tag{3.10}$$

Before releasing the output torque, saturation is applied for safety so that it never exceeds the maximum value allowed by the motor $T_{m,max}$.

3.2.6 Motor-transmission dynamic block

Knowing the torque that the motor generates and taking into account the external load applied to the mobile surface and the viscous friction, we implement the Motor-Transmission dynamic block in figure 3.12.

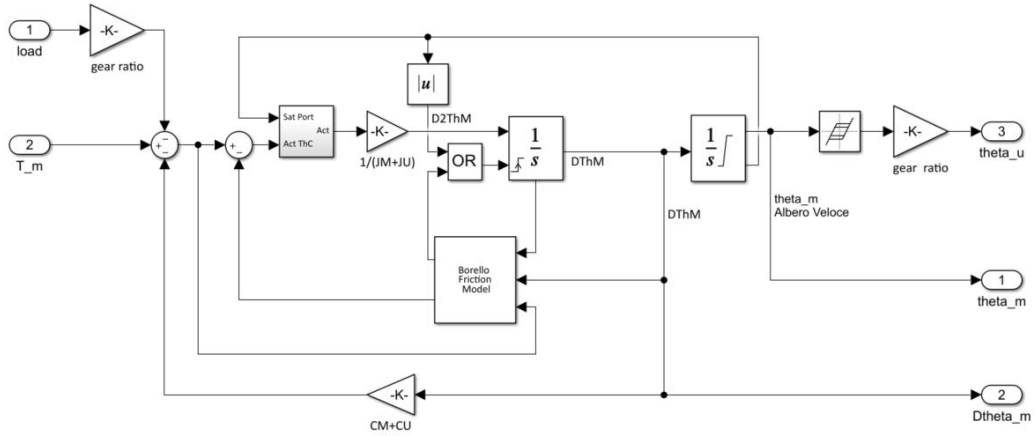

Figure 3.12: Motor-transmission dynamic block on Simulink

Through it, we will be able to calculate the torque required to execute a required position or speed command. We anticipate that the friction (T_{vf}) will be calculated taking into account the modeling proposed by the Borello's method[20]:

$$T_{vf} = (C_u + C_m)\dot{\theta}_m = C_{tot}\dot{\theta}_m \tag{3.11}$$

Where $\dot{\theta}_m$ is the angular velocity of the engine and C is the viscous damping. The resulting torque (T) will then be calculated with the following formula:

$$T = T_m - \tau Load - C_{tot}\dot{\theta}_m \tag{3.12}$$

The load is multiplied by τ (in our case set $\tau = \frac{1}{500}$) which represents the reduction ratio. The torque value, before being able to be used, must necessarily pass through a saturation block. If the reaching of a mechanical end-stop is detected, the torque value is brought to zero. More precisely this happens when the saturation port assumes the value 1 (or -1). Or when torque and saturation port have the same sign but the torque reaches a value of 0.5. In all other cases the output is just the calculated torque. Subsequently with the 3.13 we can calculate the angular acceleration $\ddot{\theta}_m$ of the engine and by integrating it twice, go back to the position of the engine itself.

$$\ddot{\theta}_m = \frac{T}{J} \tag{3.13}$$

Where J is the inertia of the motor. The last block encountered implements the mechanical backlash that can be encountered in reality, defined by a dead band centered on the output. The user position θ_u can be easily obtained by multiplying the output of this block with the transmission ratio τ.

3.2.7 Signal acquisition block

The last block that we analyze does not represent a physical component of the actuator but we need it for our work, in particular to make the comparison between the Reference Model and the Monitor Model. The comparison will be made by the fitness function (which we will describe in chapter 5) which will calculate the error between the equivalent currents of the phases of the two models. But as we have seen, the RM works by simulating the three real phases, while the MM (as we will see later) is a simplified model that works with a single phase. Hence the need to adapt the three currents of the three phases of the RM to a single equivalent current $I_{3,eq}$. In figure 3.13 we see the internal composition of the block.

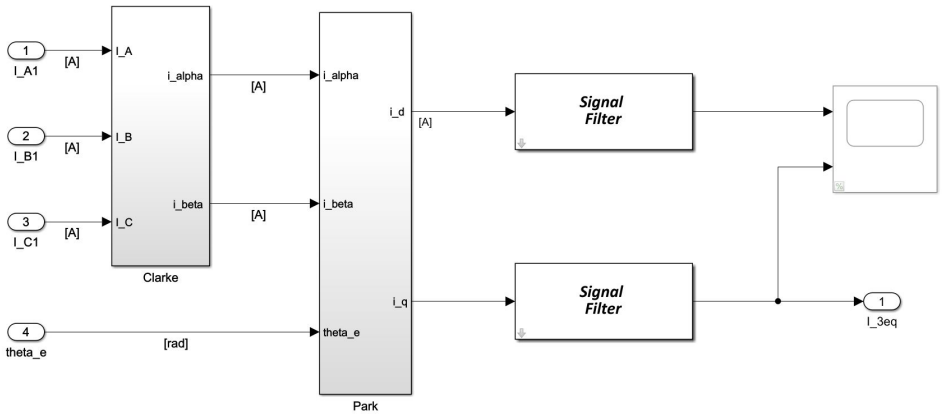

Figure 3.13: Signal acquisition block

The three currents will be transformed into the equivalent current according to the relationship 3.14 suggested by [27].

$$I_{3,eq} = I_q \cdot \hat{q} = \left[-I_A + \frac{1}{2}(I_B + I_C)\right] \sin\theta_e + \frac{\sqrt{3}}{2}(I_B - I_C)\cos\theta_e \tag{3.14}$$

The Clarke-Park transformation produce the direct current I_d and the quadrature current I_q. The current produced by the coils should always be oriented perpendicularly to the magnetic field of the rotor in terms of electrical phase, so the direc current I_d will be zero[27]. For this reason we do not see it reported in eq. 3.14. It should also be noted that the θ_e appears in the equation.

3.3 Monitor Model

3.3.1 High-level Monitor Model

In this section we will analyze the second model used. The low-fidelity model or Monitor Model. As already mentioned, it will be a simplified version of the RM, of which a certain similarity can be noted by looking at figure 3.14. The simplifications made, necessary to reduce the computational costs due to the numerous iterations, will not however make us lose too much accuracy. The main difference lies in the absence of the three-phase inverter. By operating with only one phase (single current) it will be possible to reduce the integration step by an order of magnitude. Also in the electrical module we will insert explicit corrections of certain parameters (on the active phases and on the eccentricity) to take account of possible failures. This step is due to simplification of the model, which is not needed in the reference model. After a description of its main components we will compare the outputs generated by the two models under nominal conditions.

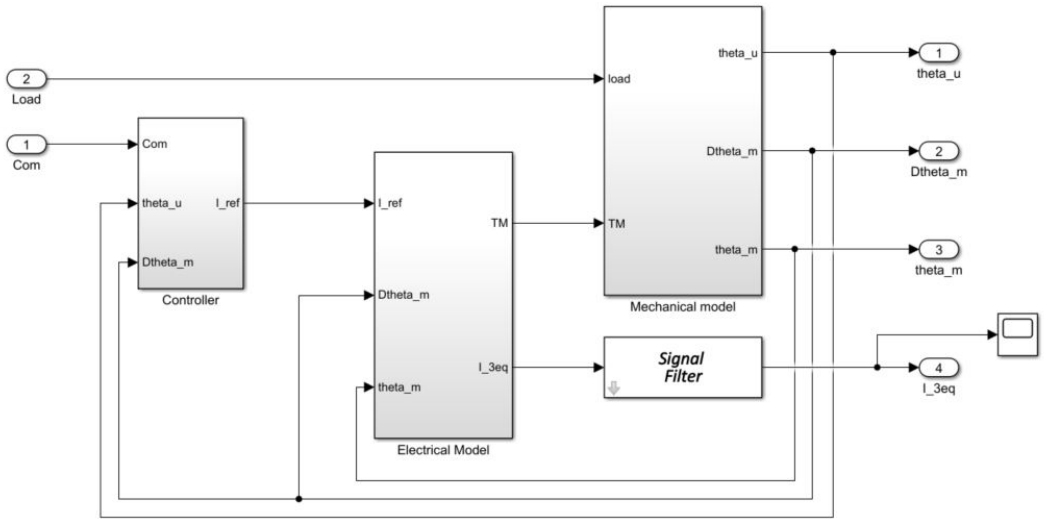

Figure 3.14: Monitor Model on Simulink

3.3.2 Controller block

The structure of this block is very similar to the one used in the Reference Model. Taken as input the command *Com* and the position of the user θ_u and the speed of the motor $\dot{\theta}_m$ the respective errors are calculated. The goal is to have the current I_{ref} in the output that will regulate the operation of the next block. Both speed and current are also subject to a saturation block to avoid values exceeding the allowed limits. The correction action, carried out in the RM by the PID, is delegated in

this case to a simple proportional gain. In figure 3.15 we have the Simulink scheme.

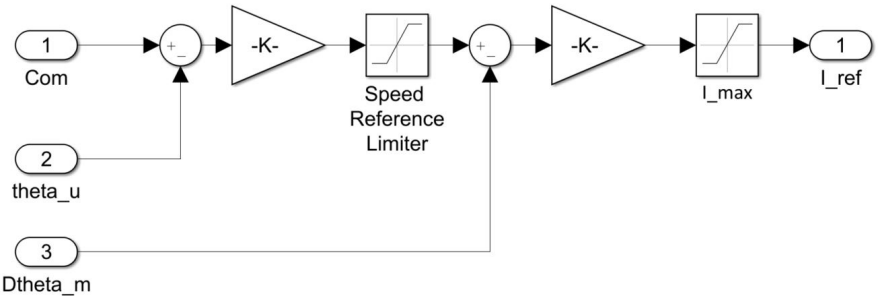

Figure 3.15: Controller block

3.3.3 Electrical model block

As mentioned before, the MM will work with a single phase, i.e. a single current. The I_{ref} proposed in output by the controller module is compared with the $I_{3,eq}$ taken as feedback. The result of this comparison is taken over by the sign module, which will multiply the sign to the supply voltage V_{max}. Summarizing we can have the following three cases:

- if $I_{3,eq} > I_{ref} \longrightarrow V = -V_{max}$
- if $I_{3,eq} < I_{ref} \longrightarrow V = V_{max}$
- if $I_{3,eq} = I_{ref} \longrightarrow V = 0V$

However, the operation described up to now only takes into account operation under nominal conditions. As mentioned earlier we also need to evaluate the possibility that failures will develop. For example, the occurrence of a short circuit will see the voltage change and further corrections must be made for any eccentricity. Even working with a single phase (single current) will vary the contribution of the counter electromotive force. At the bottom of the diagram in fig 3.16 we can see how we are going to take these three phenomenologies into consideration.

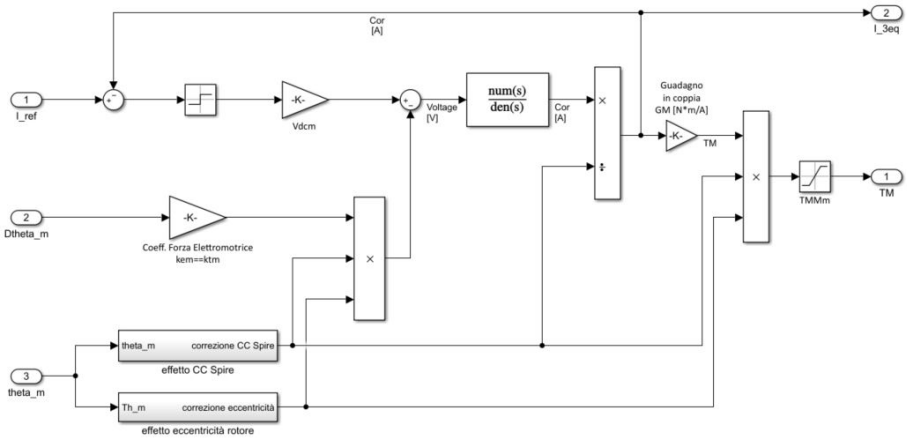

Figure 3.16: Electrical block

Now the correct voltage can be used and supplied to the motor which we see modeled by the transfer function block. The equation contained is the 3.15.

$$\frac{I}{V} = \frac{1/R_m}{\frac{N_A + N_B + N_C}{3}\tau_{RL,m}s + 1} \tag{3.15}$$

Where $\tau_{RL,m}$ is the time constant of the circuit, R_m is the resistance of the windings and L the inductance. Values are the same as those summarized in table 3.3. The 3.15 shows us how the voltage depends on the number of phases (N_A,N_B,N_C,) actually working. The result is finally the current $I_{3,eq}$ which will initially be multiplied by a torque gain of 0.0752 $\frac{Nm}{A}$ to obtain the motor torque T_m. It will still receive coil and eccentricity corrections to then finally be saturated with the maximum permissible torque values.

3.3.4 Mechanical model block

For this last block we can refer to what was said in the 3.2.6 section. In the MM the only difference lies in adopting only the viscous damping coefficient of the engine C_m.

3.4 Model outputs

Now that we have a clear understanding of the functioning and composition of the two models, we are concerned with studying how faithful the MM is, in terms of dynamic response, to the RM under nominal operating conditions. This is to verify the simplifications introduced do not lead to excessive differences in the

response. To do this, through the *Com* block of both models, we will supply a *Chirp* command as input which is the most variable over time and we will carry out a 0.5 s simulation. The values assumed by the command for the test are summarized in table 3.8. We will set the external load to zero. The parameters taken into consideration to verify the fidelity of the Monitor Model are the angular position θ_m assumed by the motor, the angular position assumed θ_u by the user, the angular speed of the motor $\dot{\theta}_m$ and the equivalent current $I_{3,eq}$. We begin to analyze the first variable θ_m through the figures 3.17.

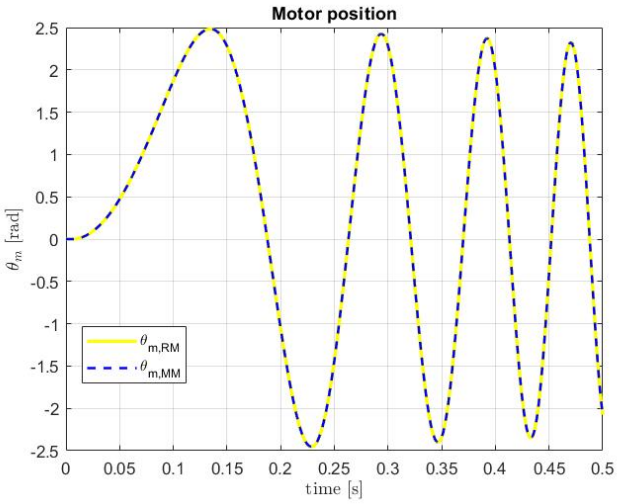

Figure 3.17: Motor position

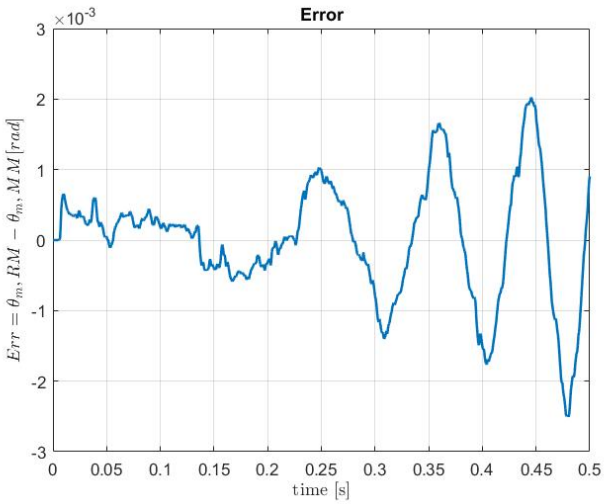

Figure 3.18: Motor position error

It is immediate to notice how apparently the two models follow the same trend without presenting differences. To confirm this, via a simple Matlab script, let's calculate the error between the two and report it in figure 3.18. Effectively, the error remains confined to a range of the order of 10^{-3} rad and has its maximum in correspondence with higher command frequencies.

We do the same analysis regarding the position of the user. Although not useful for prognostic purposes, it can provide us with further evidence of the reliability of the two models. In figures 3.19 and 3.20 we observe user position and error respectively. Also in this case the position is faithfully traced and the error is three orders of magnitude smaller than in the previous case. However, this is also due to the presence of the control electronics[26].

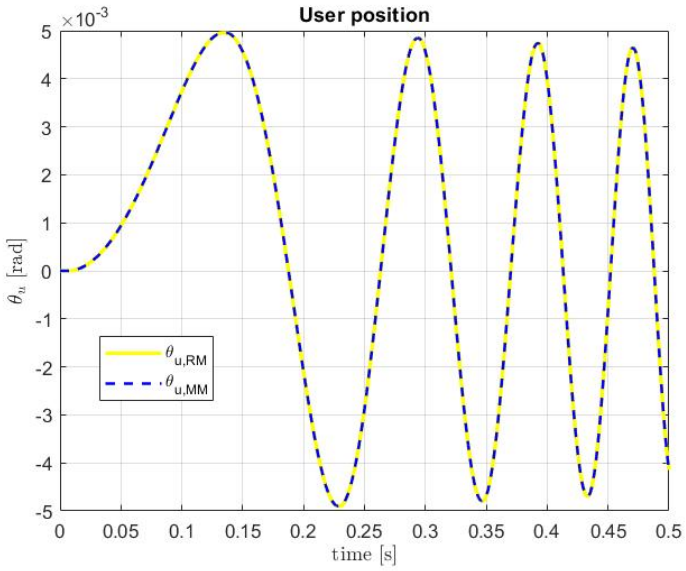

Figure 3.19: User position

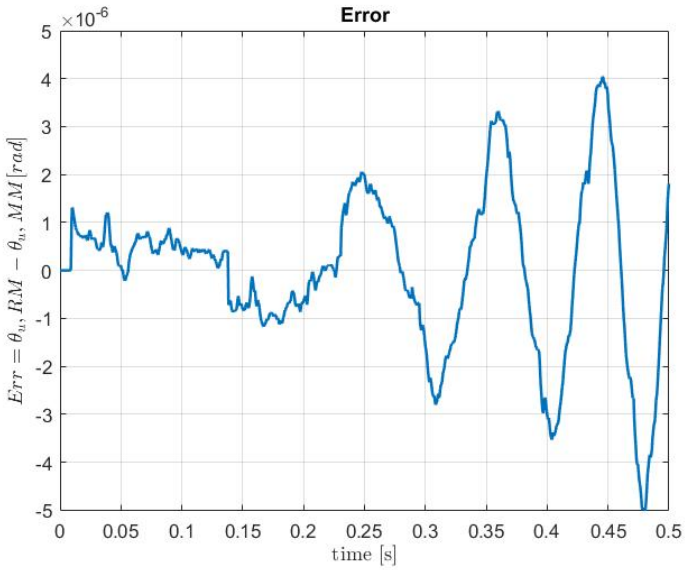

Figure 3.20: User position error

Now let's go back to the engine and evaluate the response of the models in terms of engine speed. As we expected, also in this case the response of the MM towards the RM seems to be faithful (fig 3.21). The error, although always with a negligible order of magnitude, is greater than in the previous cases, with much more rapid oscillations (fig 3.22).

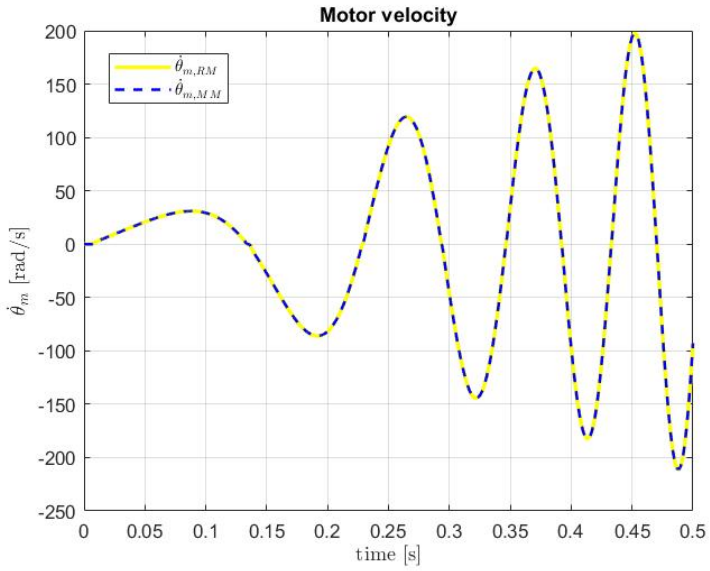

Figure 3.21: Motor speed

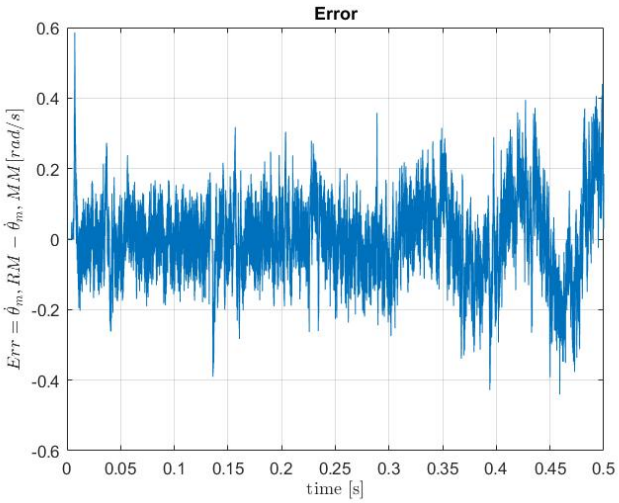

Figure 3.22: Motor speed error

The last value taken into consideration is the equivalent current $I_{3,eq}$. It is of considerable importance because it will be used as a parameter for prognostic purposes. Referring to the graphs in figure 3.23 and 3.24, we note that the error increases as the frequency of the command increases. The higher the speed at which the actuation direction change occurs, the higher the necessary torque will be required to counteract the inertia forces that develop. But referring to equation 3.10 we know that the torque is proportional to the current. The current required to develop several torques in a short time will also be subject to sudden variations, generating more errors.

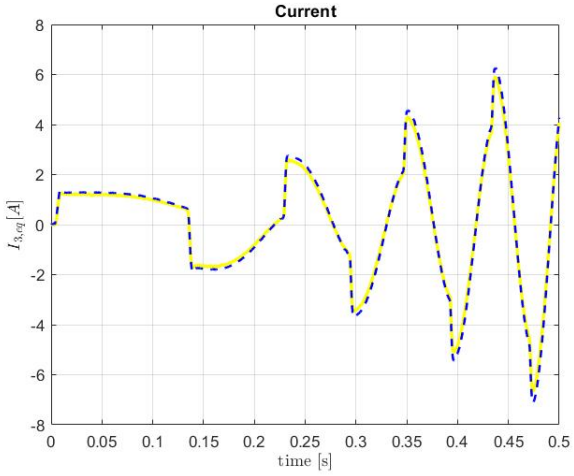

Figure 3.23: Equivalent current

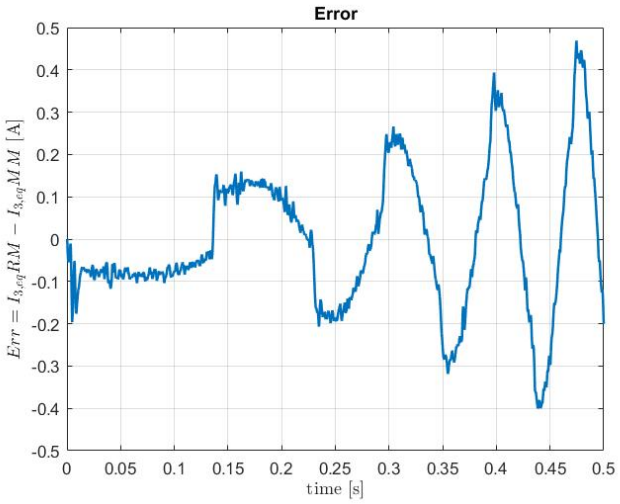

Figure 3.24: Equivalent current Error

3.5 Fault implementation

Bearing in mind what is explained in section 2.3.1 regarding the type of failure that can occur in an electromechanical actuator, we can study how these modify the dynamic response of the actuator subject to various commands. Among the many we will simulate the following eight: friction fault (F), backlash fault (B), short circuit fault (N), eccentricity fault (ζ) and proportional gain fault (G). To do this we will apply coefficients, one for each failure, to the parameters of the nominal conditions of the reference model. Furthermore, for each failure applied, the response to two different commands will be evaluated: the step command and the chirp command. We will evaluate the response in terms of user position (θ_u), motor speed ($\dot{\theta}_m$) and equivalent current ($I_{3,eq}$).

3.5.1 Friction fault

Friction is a highly non-linear phenomenon that occurs between two moving sliding parts that limits the performance of the system, as well as being difficult to control by a simple control system. Since it is not linearizable, it is necessary to study it in depth before proposing its modelling. The friction behavior varies according to the conformation of the two surfaces in contact. Each has surface roughness which interacting with each other opposes the relative motion. Furthermore, friction is strongly dependent on the sliding speed and the presence of lubricant. When the latter is not present we speak of dry friction (or Columbian). The main consequences of friction are, as mentioned, a deterioration in overall performance,

erosion of the parts in contact and an increase in temperature. The formulation that we adopt to describe on a theoretical level is the one proposed by coulomb that we see in figure 3.25.

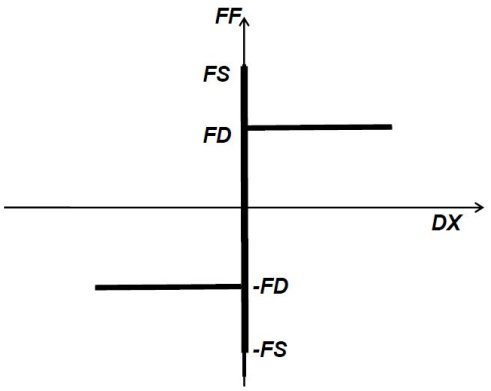

Figure 3.25: Coulomb friction model[20]

In the absence of relative speed the friction force assumes a value equal or lower than the static friction modulus (FS in the figure) while in the presence of relative motion we assume a constant value equal to the dynamic friction force (FD in the figure). We can therefore notice a discontinuity in the transition from a stationary body to a moving body which can generate instability in the control algorithms. In reality, at low speeds, the phenomenon of stick-slip occurs, i.e. a continuous and rapid transition from zero speed to speed greater than zero of the component[20]. At low speeds, to understand how the friction force varies, we can refer to the Stribeck curve in figure 3.26.

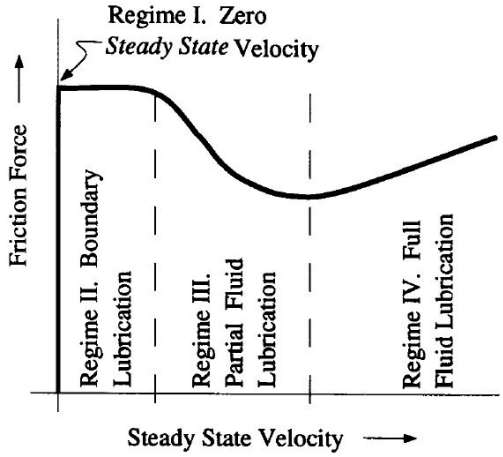

Figure 3.26: Stribeck curve[28]

Initially the component is stationary and held in place by static friction forces. The asperities joined together will only deform (in fact they are modeled as springs). This phase is called *pre-slip*. Subsequently, when the applied force exceeds the limit value FS, the body begins to move with a very low speed, not sufficient to create a state of lubricant between the two surfaces (*lubricating boundary*). As the speed increases we have a *partial lubrication* which allows the creation of a lubricating state and the decrease of the friction force. Once a critical speed has been exceeded, we enter the regime of *full lubrication* following the typical dynamics of viscous friction, i.e. with a constant slope[20].

Many modeling philosophies are available to best represent this phenomenon (Quinn, Dahl, Karnopp model etc..). However, these refer to mathematical models that, although more powerful than any purely linear representations, still suffer from some shortcomings. In fact, they still have limitations in the event of the start of the mechanical system or inversion of the sign of the speed (it requires that the speed assumes a value equal to zero for at least an instant of time). To avoid problems in the time domain simulation we present an innovative method proposed by prof. Lorenzo Borello[20]. The model has the following strengths:

- It takes into account the sign of the friction as a function of the direction of the speed;
- Effectively separates static friction force conditions (FSJ) from dynamic ones (FDJ);
- It evaluates the possible stopping of a moving component and vice versa;
- keeps the component stationary or in motion, which results respectively in adherence or dynamic conditions;
- Takes into account any limit switches that the component can hit inelastically;

The mathematical formulation of the model [20] is as follows:

$$FF = \begin{cases} -F_{att} & if \quad V = 0 \quad or \quad |F_{att}| \leq FSJ \\ -sgn(F_{att}) \cdot FSJ & if \quad V = 0 \quad or \quad |F_{att}| > FSJ \\ -FDJ & if \quad V \neq 0 \end{cases}$$

Where FF represents the friction force actually calculated and F_{att} is the active force applied to the system.

To avoid instability problems due to the change of actuation direction and therefore a temporary stop of the system, the following formulation is proposed:

$$\dot{\theta}_m(t_{i+1}) = 0 \quad if \quad \dot{\theta}_m(t_{i+1})\dot{\theta}_m(t_i) \leq 0 \tag{3.16}$$

In figure 3.12 we can see the Borello model implemented to take into account friction. Now we can see how the electromechanical actuator has a different dynamic response as a function of different friction values. In particular we will assume the static and dynamic friction coefficients starting from the nominal value (NF) up to three times the nominal value. As regards a unitary step command we have the following results:

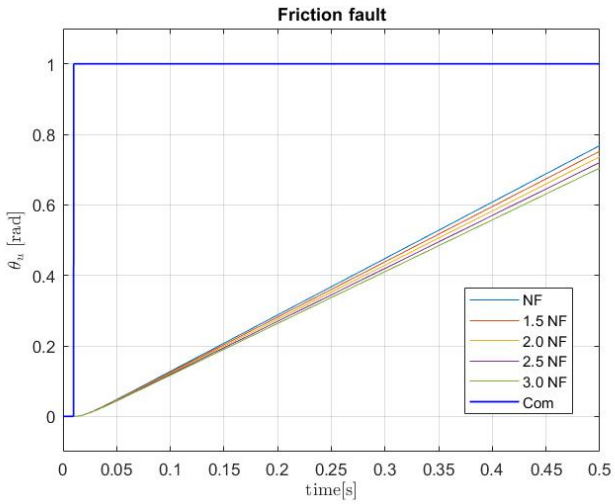

Figure 3.27: User position for a step command

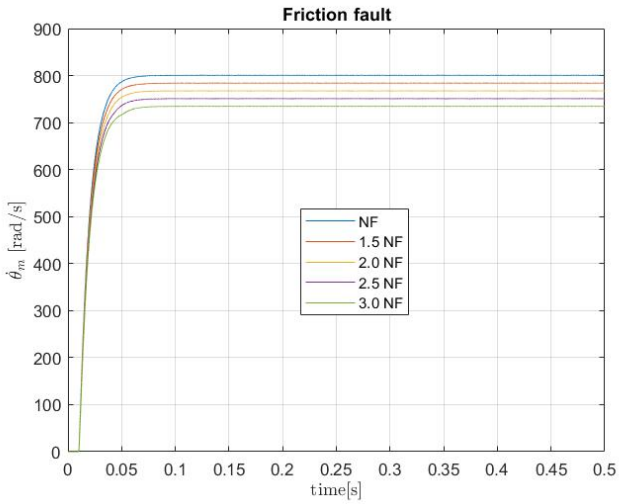

Figure 3.28: Motor velocity for a step command

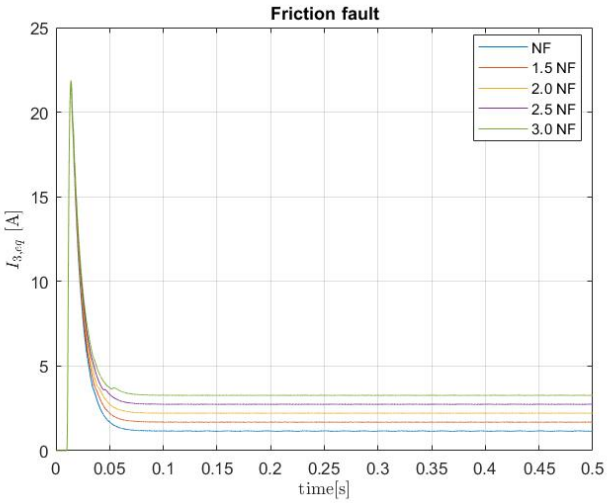

Figure 3.29: Equivalent current for a step command

As anticipated, system performance tends to deteriorate as the friction coefficient increases. We can see it in figure 3.27 and 3.28 which respectively show how the user reaches the desired position in more time than the nominal conditions and how the maximum motor speed is limited. The current $I_{3,eq}$ also varies as a function of friction. Observing the graph in figure 3.29, compared to the nominal conditions, it will be greater to counteract the greater resistant torque produced by friction. As a consequence we have a greater overheating of the windings and a greater consumption of energy. Let's now do the same job but taking a chirp command as input. We get the following results:

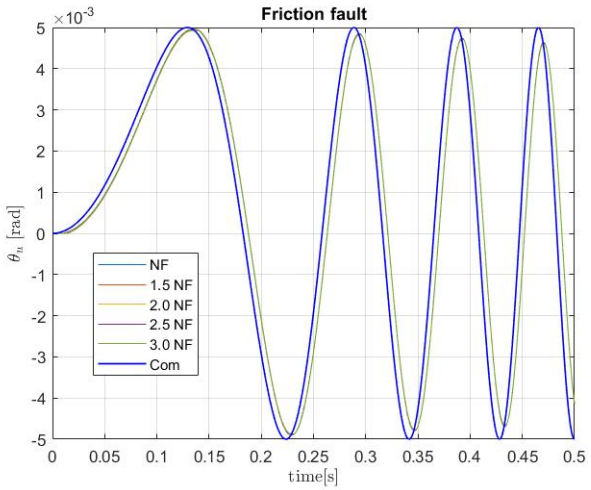

Figure 3.30: User position for a chirp command

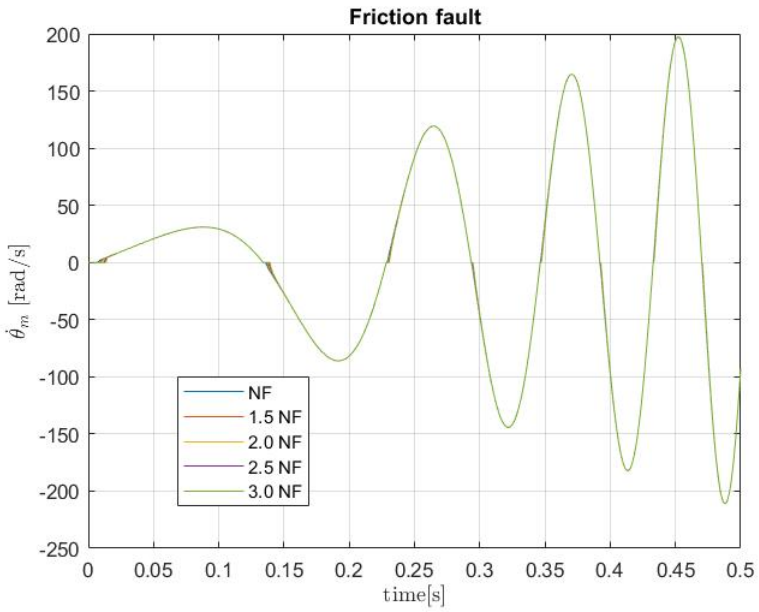

Figure 3.31: Motor velocity for a chirp command

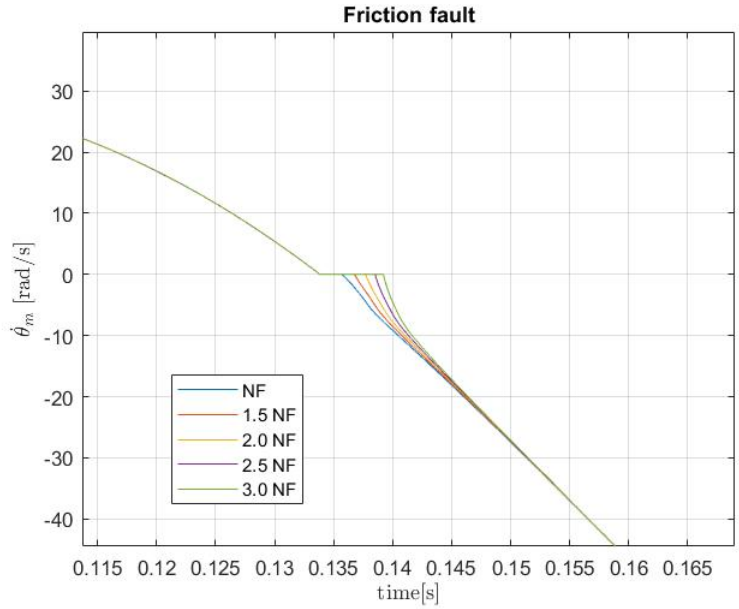

Figure 3.32: Zoom of motor velocity for a chirp command

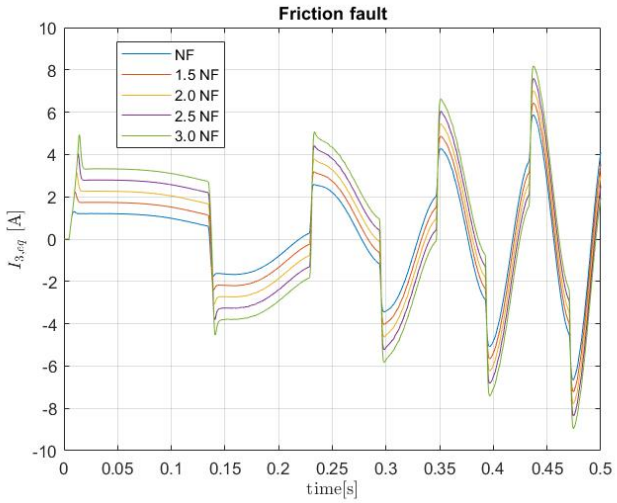

Figure 3.33: Equivalent current for a chirp command

Figure 3.30 indicates that the user's position is not particularly affected by friction. Comparing it with figure 3.31, we note that the difference in the response becomes more evident when the motor speed is zero, i.e. in the change of the actuation direction (in figure 3.32 we can see the detail). The adhesion time between the two sides increases with the increasing of static friction force. Indeed, to be able to win it and restart the system it will be necessary to develop more torque and this will require more time. We also know that the torque is directly proportional to the current supplied, therefore by observing the graph in figure 3.33 we note that the current peaks $I_{3,eq}$ are at the instants where $\dot{\theta}_m = 0$. Once passed in conditions of dynamic friction (which is less than static friction) the value of the current decreases and stabilizes. Finally, the curve will shift towards higher current values as the friction values increase.

3.5.2 Backlash fault

The mechanical Backlash looks just like an empty space between the two faces that must come into contact. We can find it, for example, between the teeth of two gears, in bearings, in mother-screw gear couplings, in ball screws. Given the impossibility in the production processes of obtaining components without imperfections, in the union of two parts we will always experience the phenomenon of backlash. Furthermore, the contact between two components (especially if sliding) will cause them to wear, increasing the backlash over time. However, in many applications, the presence of this space, which is not excessively large, will be necessary to deal with various situations such as, for example, thermal expansion and the need to provide a lubricating layer between the two parts.

Like friction, backlash also carries with it non-linear phenomena in the dynamic response. In particular, as we shall see, the major problems arise in the direction changes of the actuator. An excessive increase beyond the expected limits leads to a gradual worsening of the expected response. The main effects that we can see are vibrations and inaccuracy in reaching the desired control position.
In our work we will take this effect into account in both models by inserting the backlash block. It inserts a dead band (DB) whose width specifies the amount of play present. It is quantified in Simulink with the equation 3.17.

$$DB = \frac{BLK}{\tau} = \frac{10^{-5}}{\frac{1}{500}} = 5 \cdot 10^{-3} \tag{3.17}$$

We will simulate the failure with different Backlash intensities (2, 5, 10, 50, 100) with respect to the nominal conditions. Let's start by evaluating a step-type input. In the figures below we observe how the three parameters θ_u, $\dot{\theta}_m$ and $I_{3,eq}$ taken into consideration evolve over time.

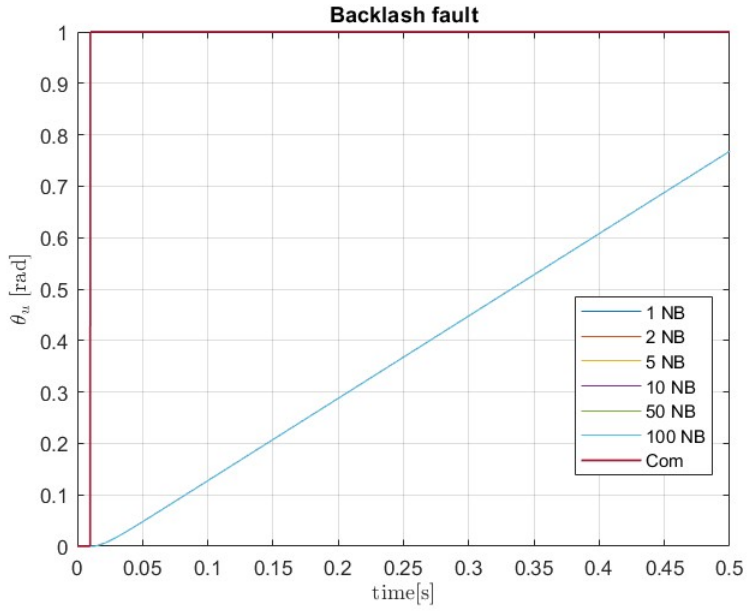

Figure 3.34: User position for a step command

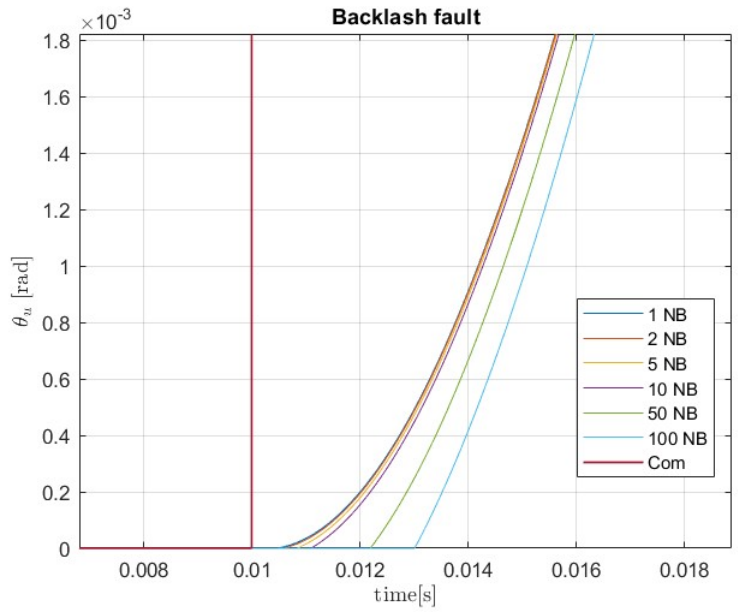

Figure 3.35: Zoom of user position for a step command

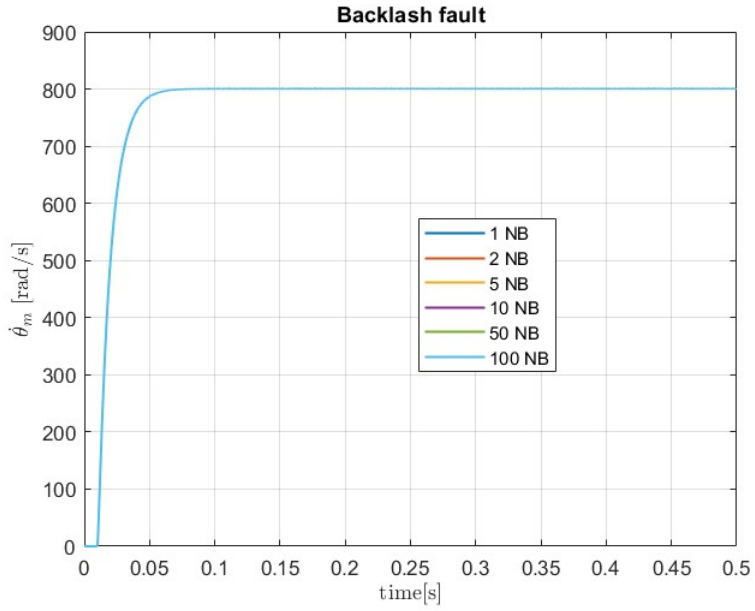

Figure 3.36: Motor velocity for a step command

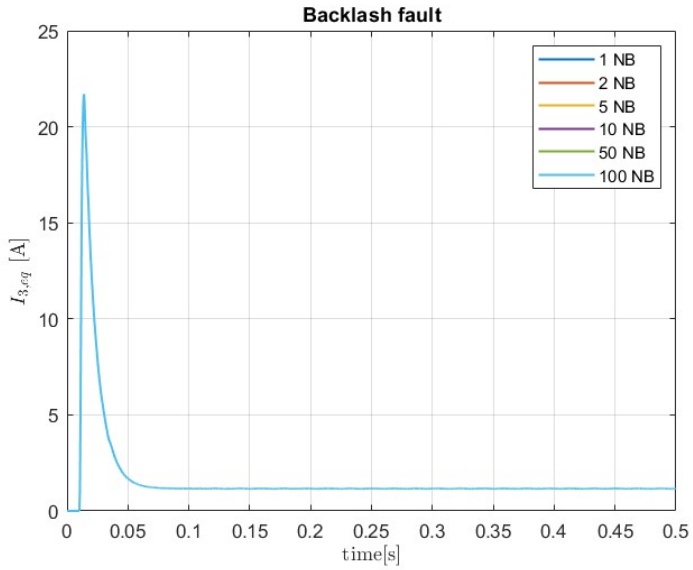

Figure 3.37: Equivalent current for a step command

The backlash does not present particular criticalities as regards a step input as there are no changes in direction. As the backlash increases, we note in figure 3.35 an increase in the response delay.

Let's now simulate the more complex chirp command and evaluate the response summarized in the following graphs.

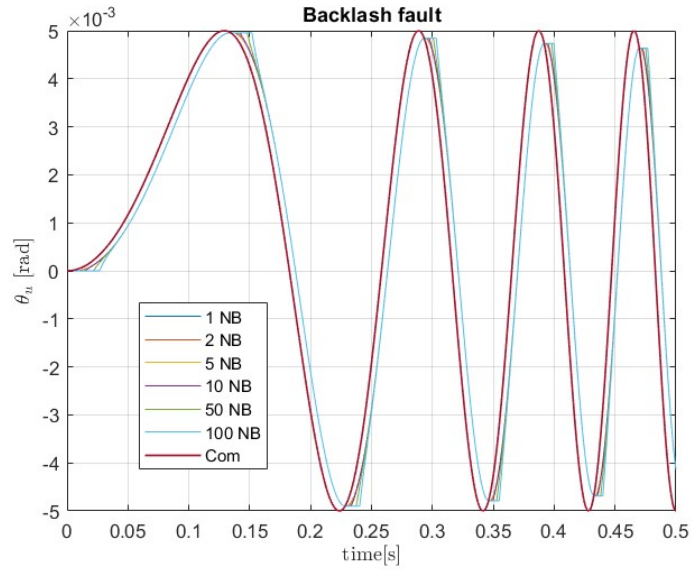

Figure 3.38: User position for a chirp command

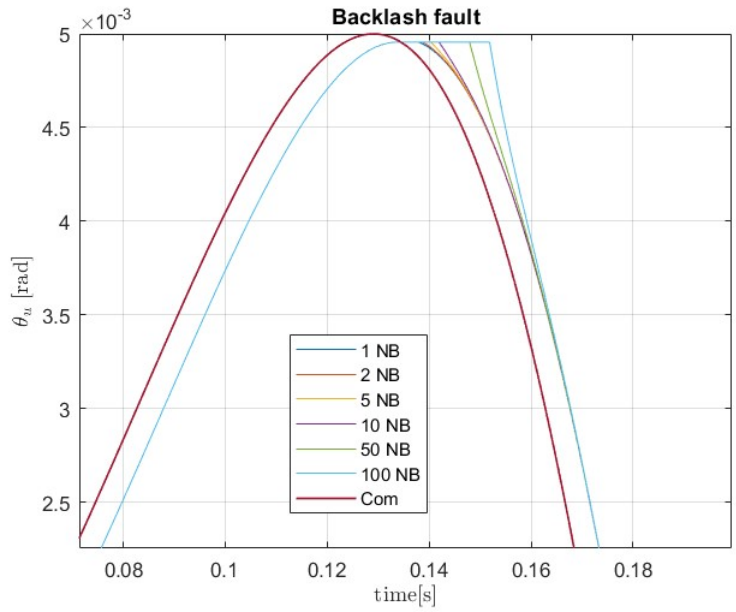

Figure 3.39: Zoom of user position for a chirp command

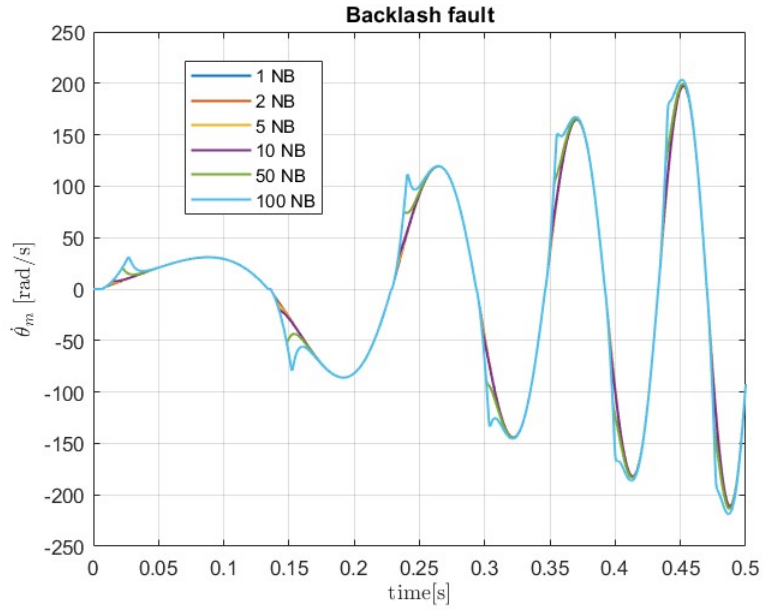

Figure 3.40: Motor velocity for a chirp command

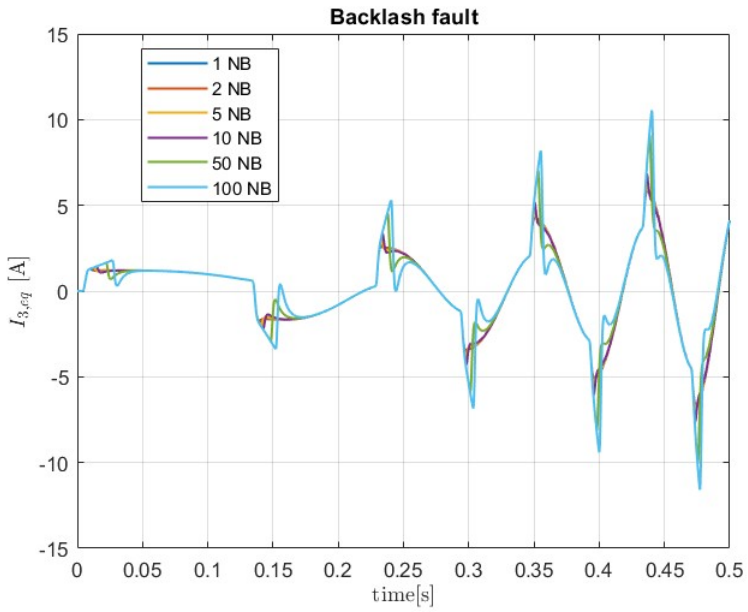

Figure 3.41: Equivalent current for a chirp command

As we anticipated earlier, the main problems arise in the presence of a change in direction of the actuator. In fact, we note in figure 3.39, an increase in the delay of the response in correspondence with these points. Error that tends to decrease as the command frequency increases. Consequently, the engine speed (fig 3.40) will also be affected at the above points. Having ascertained that an increase in speed is strictly correlated to an increase in the equivalent current, we note in figure 3.41 that as the intensity of the backlash increases, increasingly larger current peaks occur. These, if taken beyond the permitted limits, can cause damage to the windings and the motor from overheating.

3.5.3 Short circuit fault

In the last twenty years the application of BLDC motors has experienced a remarkable development in various fields thanks to their characteristics of being able to work in hostile conditions and high temperatures. Winding-related failures are the most frequent and if diagnosed early enough it is possible to avoid catastrophic failures. Of these failures those of the stator represent 30-40% of the total and are caused by a deterioration of the insulating layer (80% of stator failures) which cause short circuits[29]. The figure 3.42 summarizes the most common types of failures that can occur in the stator.

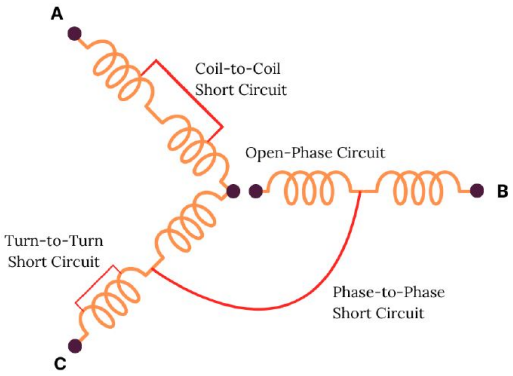

Figure 3.42: Common BLDC motor stator failure [29]

The short circuit itself occurs when two parts with low resistance or impedance are brought into contact with each other allowing the flow of an overcurrent called short circuit current. It will increase the temperature of the windings, which is already high in certain applications. In our case we will deal with studying a coil-coil type short circuit because it is the only one which, when it occurs, allows the motor to continue running for a certain period of time[29][26].

The short circuit can be identified by evaluating the variation of the external magnetic field according to the following equation:

$$K_{E,i} = \frac{\partial \phi}{\partial \theta_m} = NA \frac{\partial (\iint_A B \hat{n}\, dS)}{\partial \theta_m} N_i \tag{3.18}$$

The equation 3.18 is the compact form of the 3.3, 3.4 and 3.5. For the generic phase indicated by the letter i, we impose a failure percentage level using the coefficient N_i. It will assume the value 1 (100%) in nominal conditions. This means that all the windings are not affected by a short circuit. In our work we will simulate the short circuit for phase C (N_C) with the following failure percentages: 0%, 25%, 50%, 75% and 100%. Numerically, however, when we simulate a 100% short circuit we will set $N_C = 10^{-6}$ to avoid errors. As usual, let's see what the dynamic responses are by applying a step command and a chirp command.

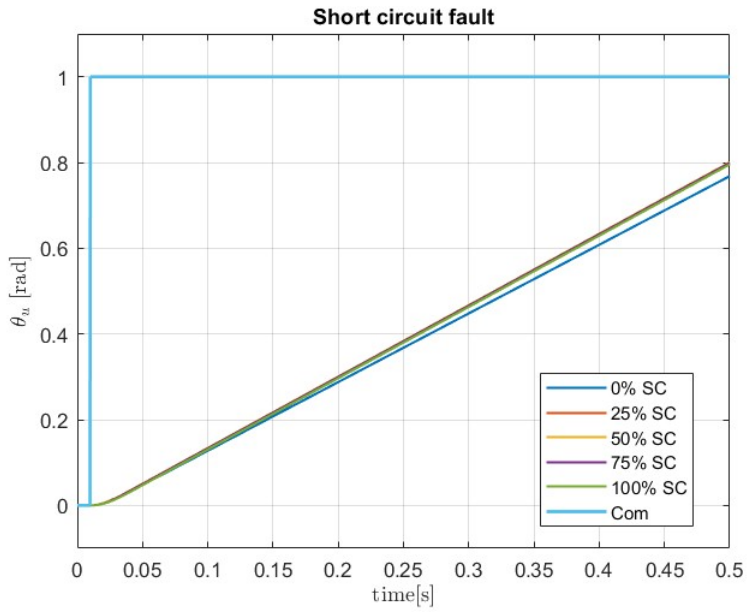

Figure 3.43: User position for a step command

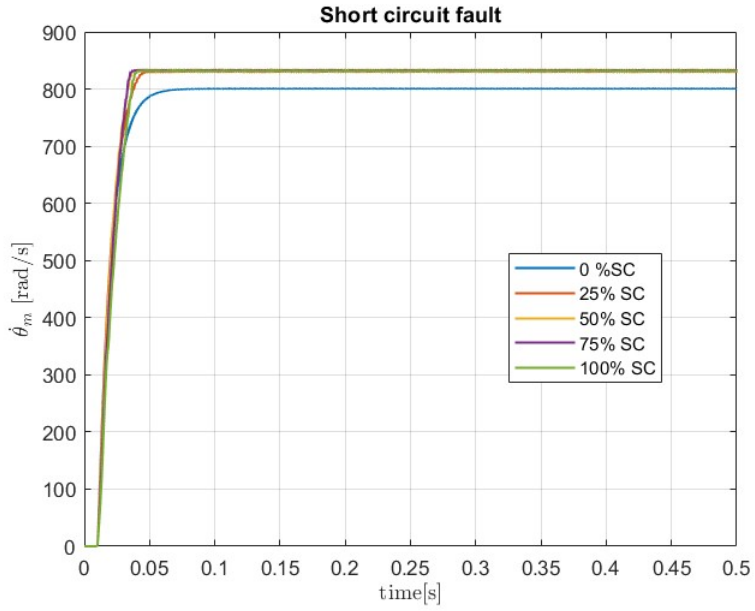

Figure 3.44: Motor velocity for a step command

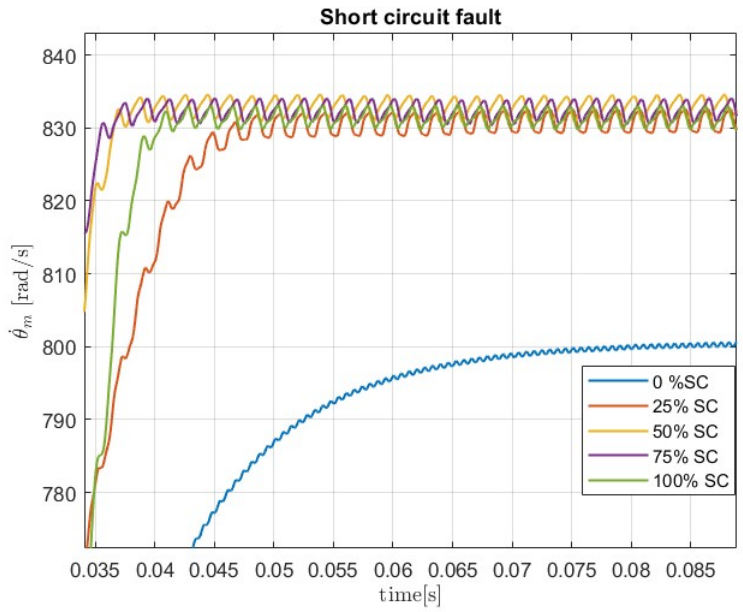

Figure 3.45: Zoom of motor velocity for a step command

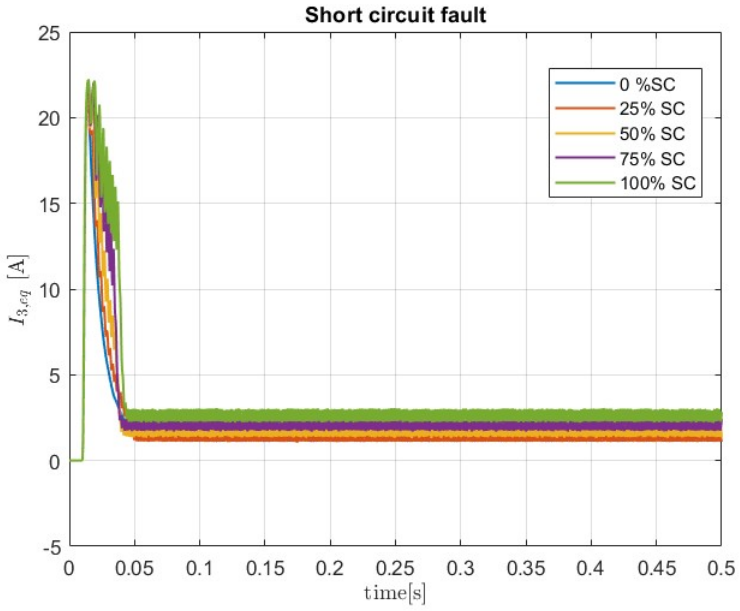

Figure 3.46: Equivalent current for a step command

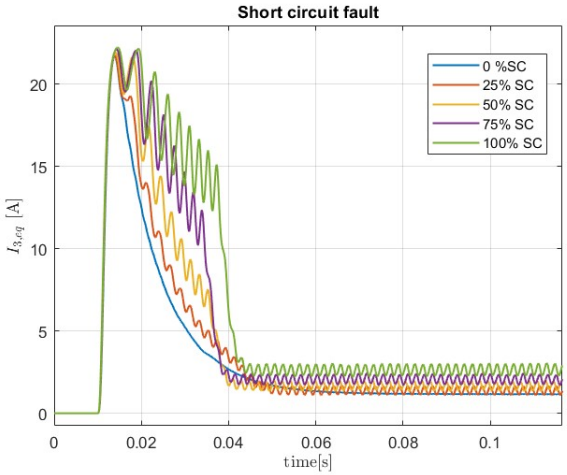

Figure 3.47: Zoom of equivalent current for a step command

First of all we can note that the introduction of a short-circuit failure causes oscillations in the equivalent current and consequently in the motor speed. In confirmation of what has been said above, figure 3.46 shows us how the primary effect of the short-circuit is the increase in current intensity. It causes an increase in engine speed. Let's now analyze the behavior caused by an input of type chirp. Referring to the graphs in figure 3.48 and 3.50, there are no particular deviations from the nominal conditions. The predominant effect is the current variations (fig.3.52) with increasingly greater peaks as the percentage of failure increases when phase C is activated. Furthermore, they increase with increasing command frequency.

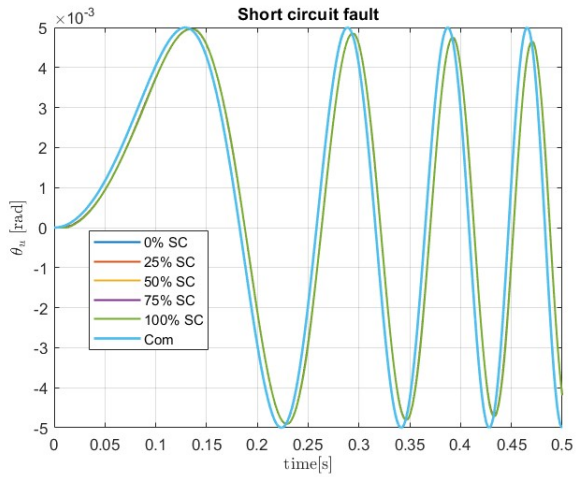

Figure 3.48: User position for a chirp command

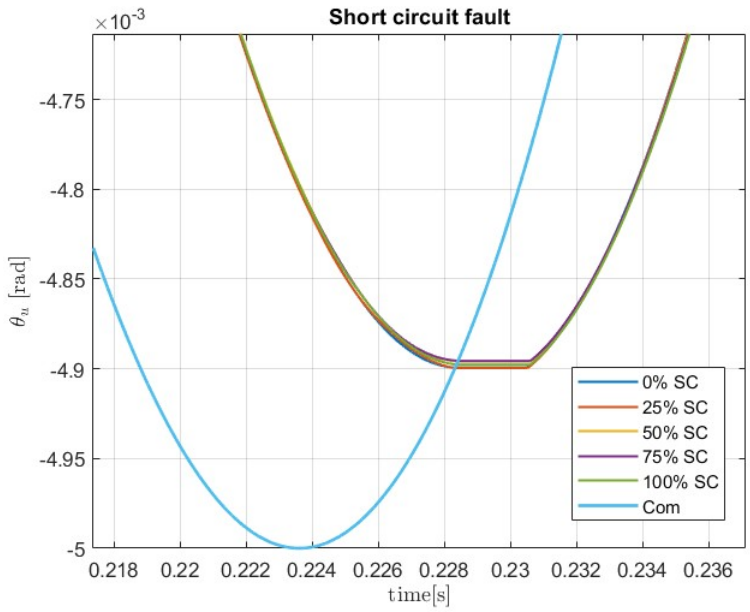

Figure 3.49: Zoom of user position for a chirp command

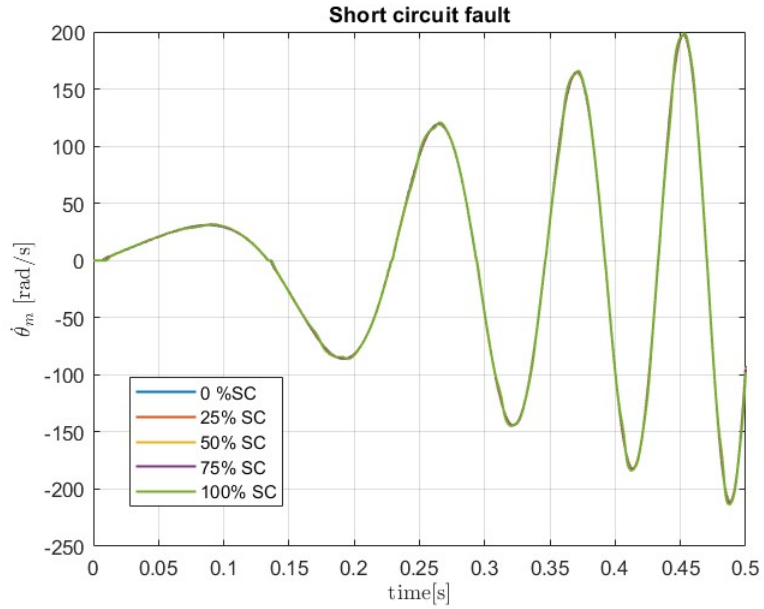

Figure 3.50: Motor velocity for a chirp command

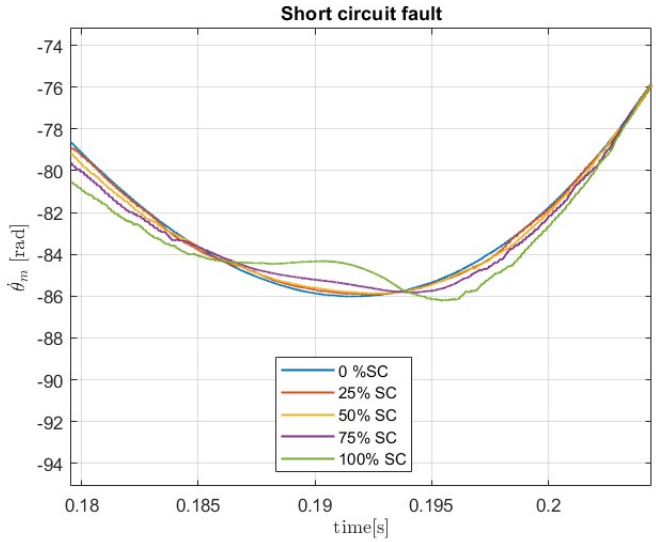

Figure 3.51: Zoom of motor velocity for a chirp command

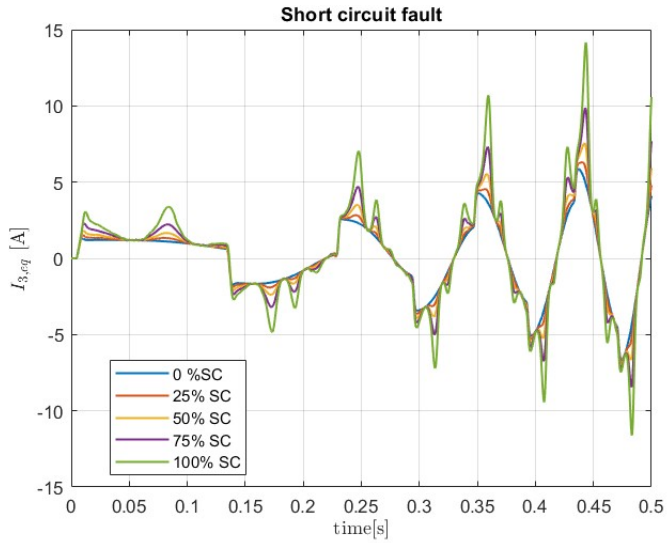

Figure 3.52: Equivalent current for a chirp command

3.5.4 Eccentricity fault

We can describe eccentricity as the mechanical failure that occurs when the stator and rotor axes do not coincide. The rotor, therefore, rotates around a different axis and the air gap with the stator is not constant. There are two types of eccentricity:

static and dynamic. In the first, the rotor rotates around its own axis which is not concentric with the axis of the stator. In the second, the rotor rotates around the stator axis. In this work we will study the static one because the dynamic one would require an analysis of the vibrations[30]. It is important to note that eccentricity can occur over time due to wear of the components even assuming an initial state of the system without imperfections. In figure 3.53 we see represented what has been described above.

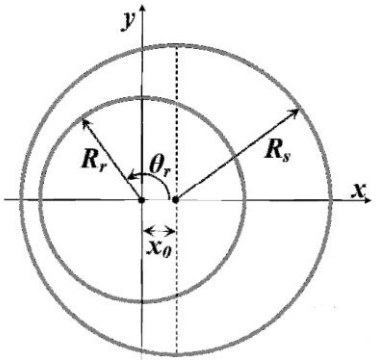

Figure 3.53: Rotor static eccentricity ζ[30]

We have already encountered the eccentricity modulus when we discussed the engine block in paragraph 3.2.5. We report its definition again:

$$\zeta = \frac{x_0}{g_0} = \frac{x_0}{R_s - R_r} \tag{3.19}$$

It is clear how an inconstant air gap, which thanks to the work done in [26] we define via 3.20 as a function of the angular position of the rotor, can cause a periodic variation (in particular sinusoidal [30]) of the magnetic field and therefore of the force acting on the rotor. This periodicity causes vibrations on the system which can lead to catastrophic failure.

$$g(\theta_r) = g_0[1 + \zeta \cos \theta_r] \tag{3.20}$$

The different dynamic responses as a function of different failure levels will be evaluated by varying the ζ coefficient from 0 to 100. The two extremes respectively represent the nominal conditions and the maximum eccentricity that can occur, i.e. the contact between stator and rotor. Let's see below what is the response to a step command.

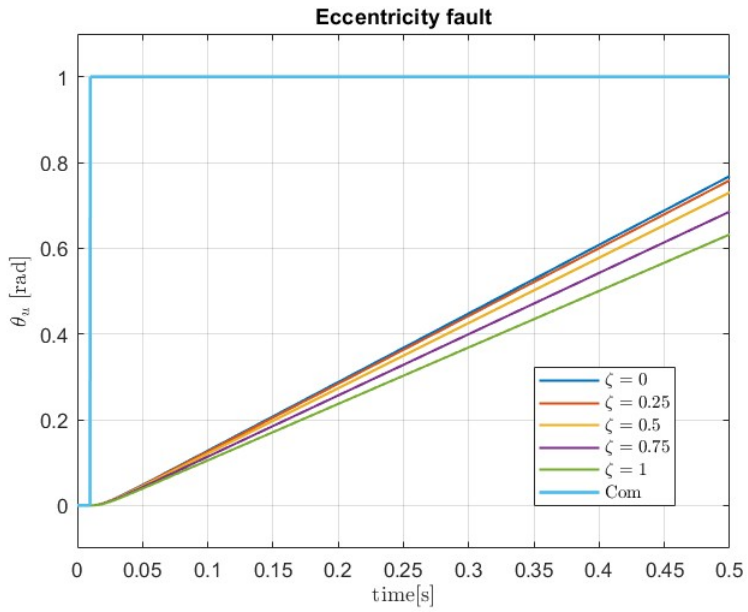

Figure 3.54: User position for a step command

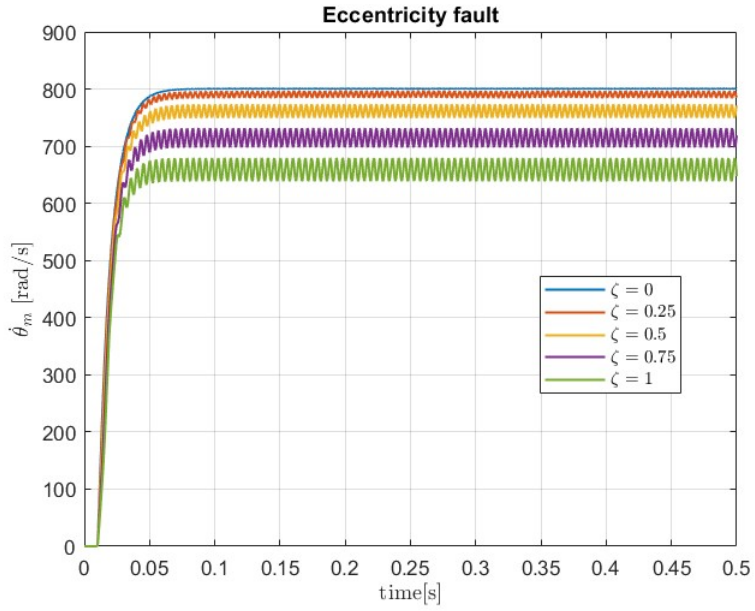

Figure 3.55: Motor velocity for a step command

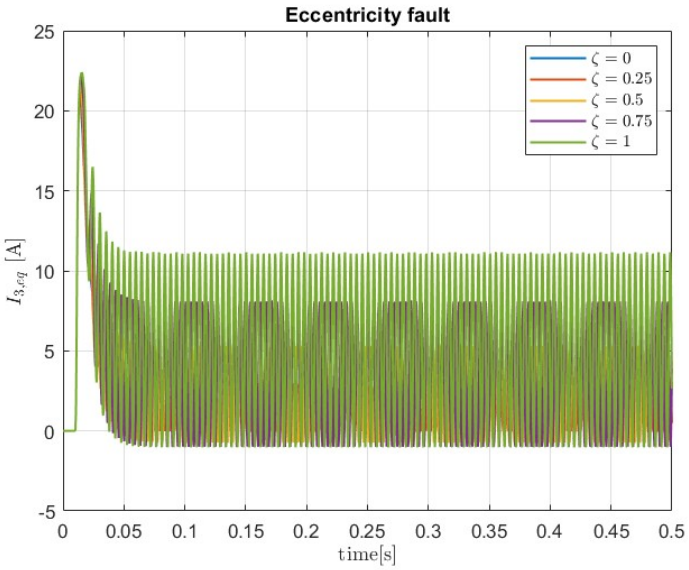

Figure 3.56: Equivalent current for a step command

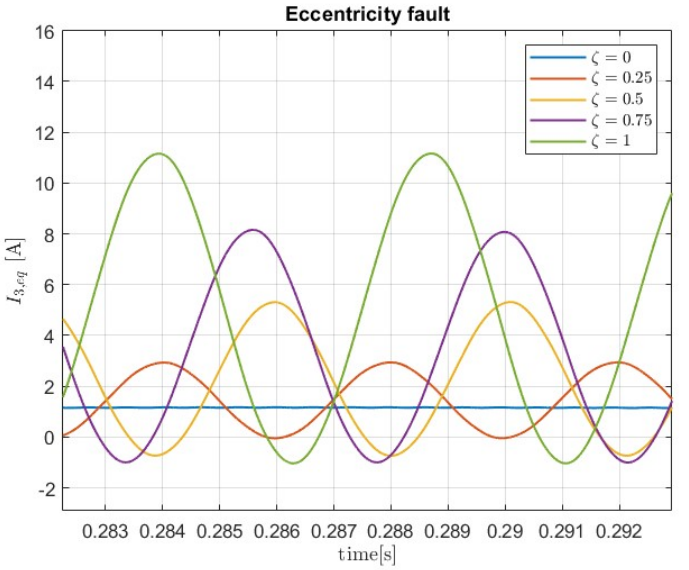

Figure 3.57: Zoom of equivalent current for a step command

Observing figures 3.54 and 3.55 we notice an effect opposite to that caused by the short circuit. In fact, as the level of eccentricity increases, the motor slows down and therefore the desired position will also be reached with a certain delay. Now studying the graph of the equivalent current in figure 3.57, we note that as

the failure rate increases, the amplitude of the oscillations increases. This is due to the fact that the asymmetrical interaction of the rotor with the magnetic field, caused by a different air gap, must be compensated with higher current to reach the same speed levels of the motor.

Let's now move on to a chirp-type command and analyze the behavior of the system. Below are the graphs of the user's position (fig. 3.58), motor speed (fig. 3.60) and equivalent current (fig. 3.62).

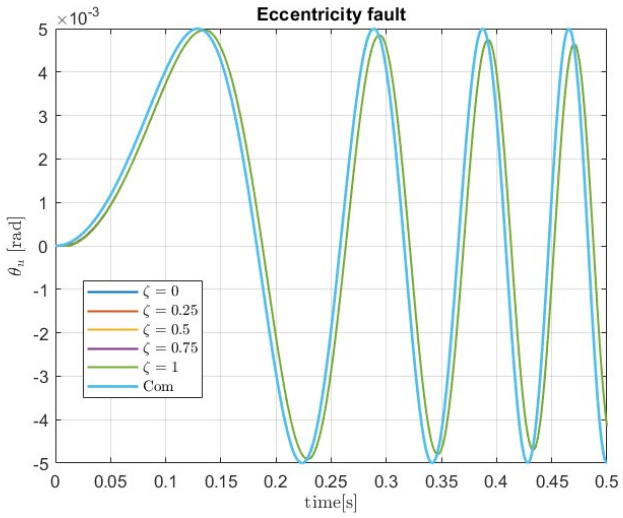

Figure 3.58: User position for a chirp command

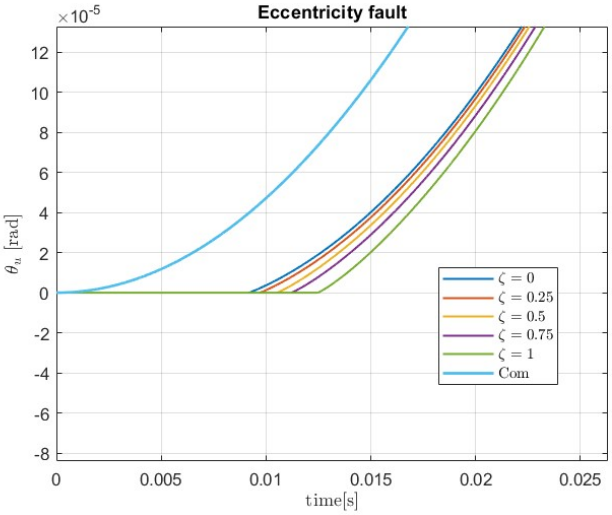

Figure 3.59: Zoom of user position for a chirp command

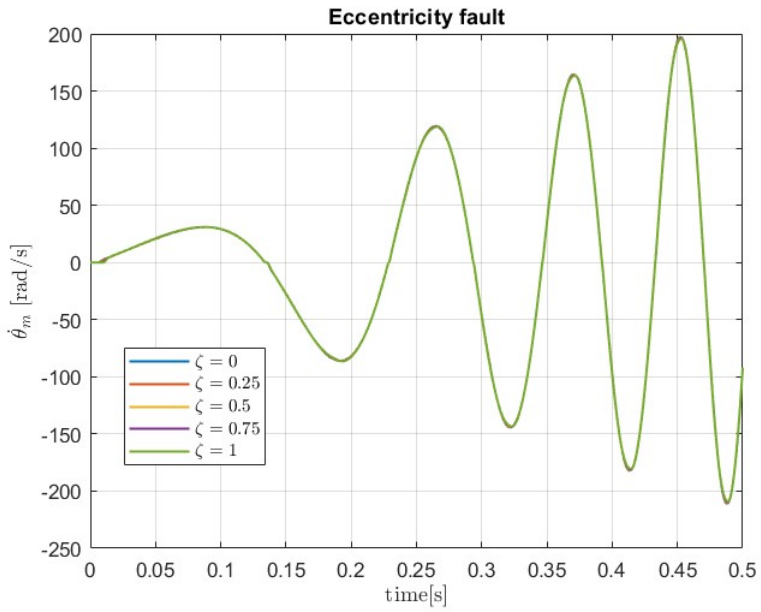

Figure 3.60: Motor velocity for a chirp command

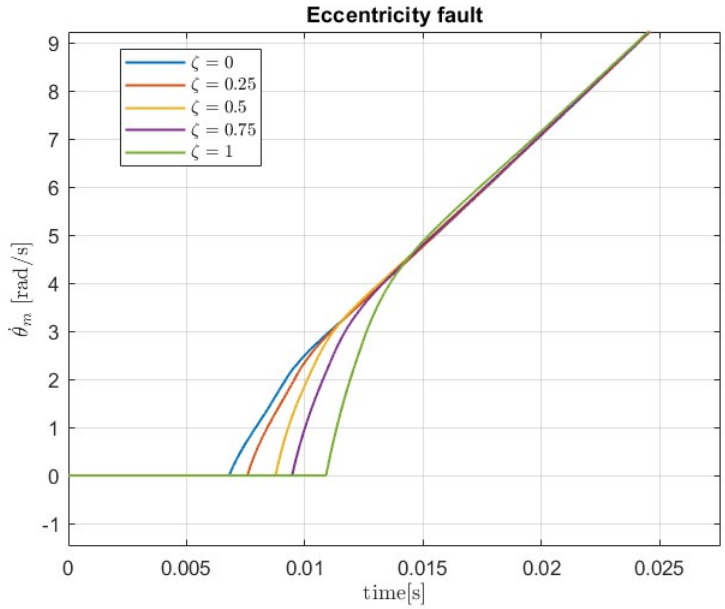

Figure 3.61: Zoom of motor velocity for a chirp command

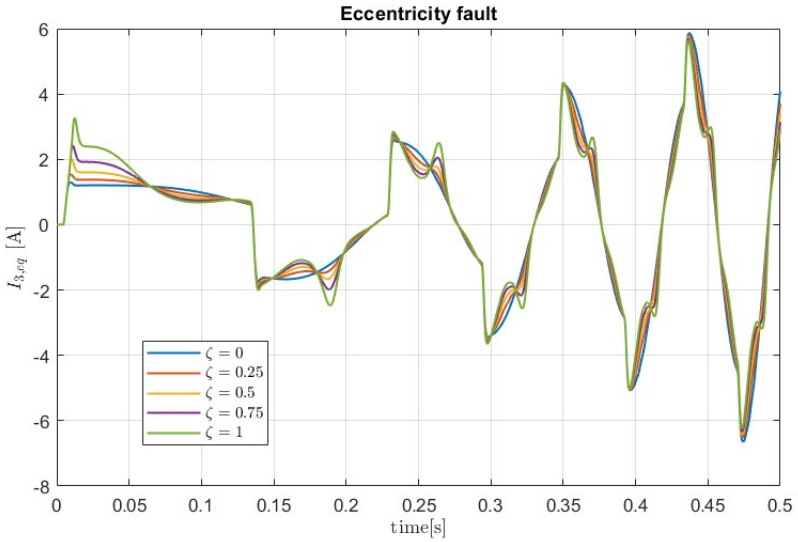

Figure 3.62: Equivalent current for a chirp command

As seen in the case of the short circuit, the problems involving a chirp type input manifest themselves in the change of direction of actuation or a change in the sign of the speed. The actuator responds with a greater delay the higher the failure level. Effect that tends to weaken with increasing speed. The equivalent current instead follows roughly what happens with a step command. The control electronics try to bring the motor back to the desired position and this can be understood by looking at the current peaks in conjunction with the activation of the phase in question. The peak current also increases with increasing failure level.

3.5.5 Proportional Gain fault

In chapter 1 we understood how much the electrification of the various subsystems and the increase of electronic components has been and will be predominant in the coming years. We have also said that a failure of these components, unlike for example one deriving from friction, occurs without premonitory signals, making their prognosis difficult. Furthermore, as the number of these increases, the probability that a malfunction may occur also increases. Referring to our case, from the diagram in figure 1.7, we know that an EMA is composed of an actuator control unit. It contains the control electronics described in paragraph 3.6. In order to study and simulate a response of the system affected by failure of this type, we will vary the proportional gain G_{prop} in a range that varies from $0.5 \cdot G_{prop}$ to $1.5 \cdot G_{prop}$. We remind that in the nominal conditions $G_{prop} = 1 \cdot 10^5 \frac{1}{s}$. The results for a step command are shown below.

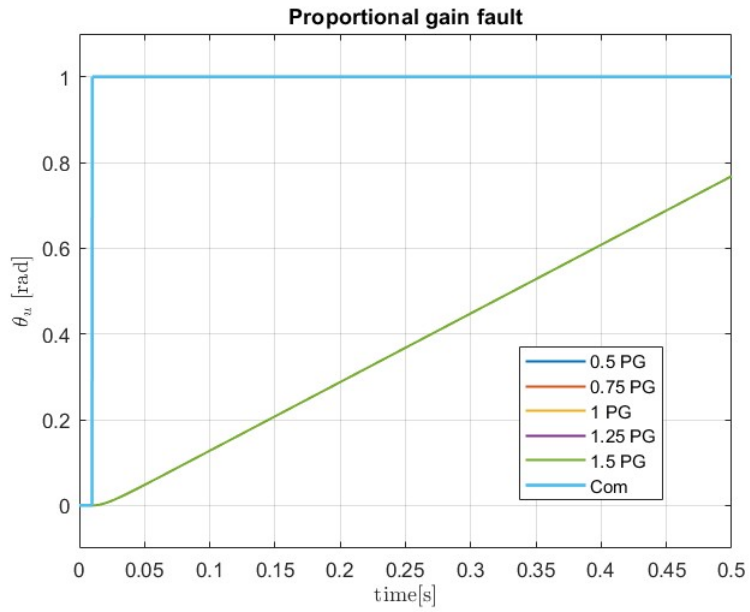

Figure 3.63: User position for a step command

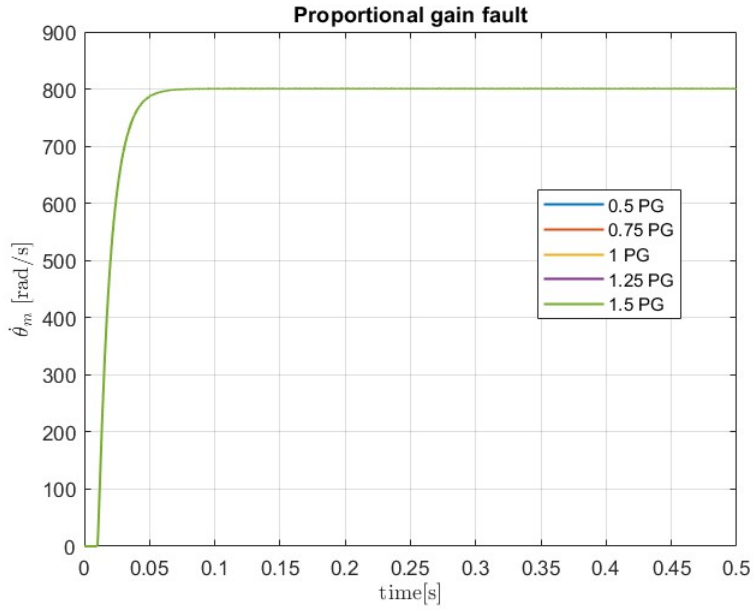

Figure 3.64: Motor velocity for a step command

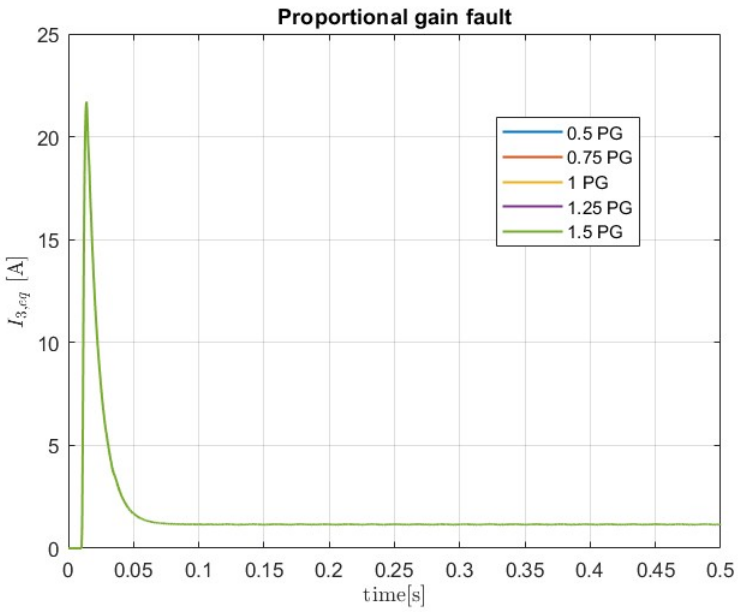

Figure 3.65: Equivalent current for a step command

It is clear that a gain fault has no effect by holding a step command as input. So let's try to simulate a response to the more complex chirp command. The results are expressed below.

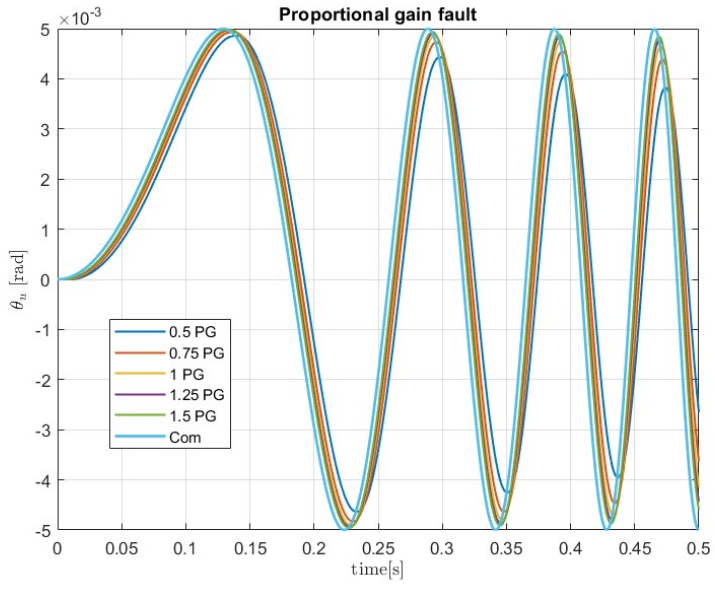

Figure 3.66: User position for a chirp command

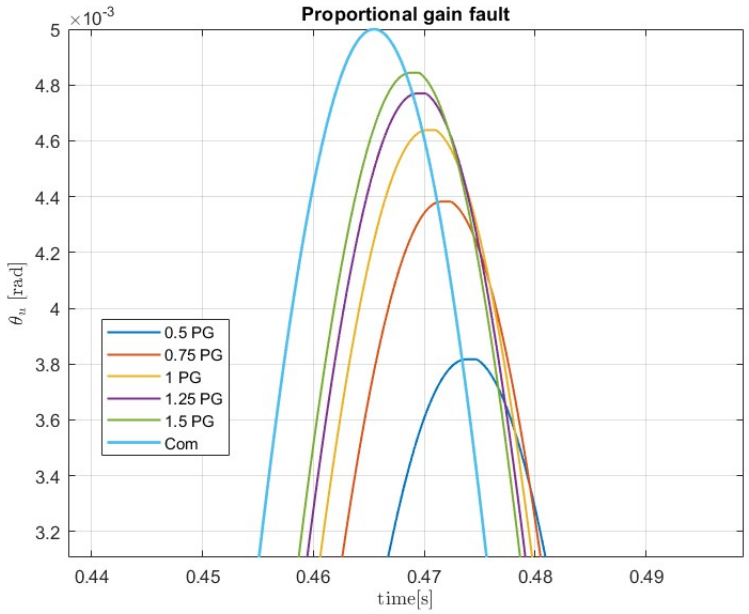

Figure 3.67: Zoom of user position for a chirp command

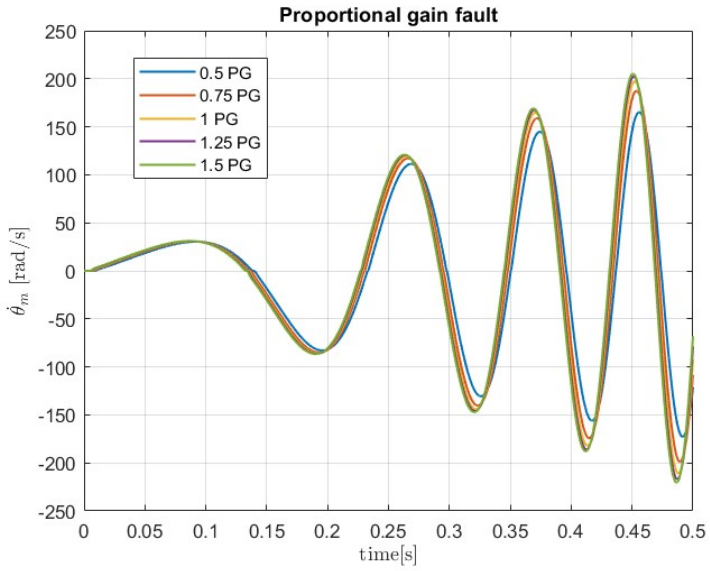

Figure 3.68: Motor velocity for a chirp command

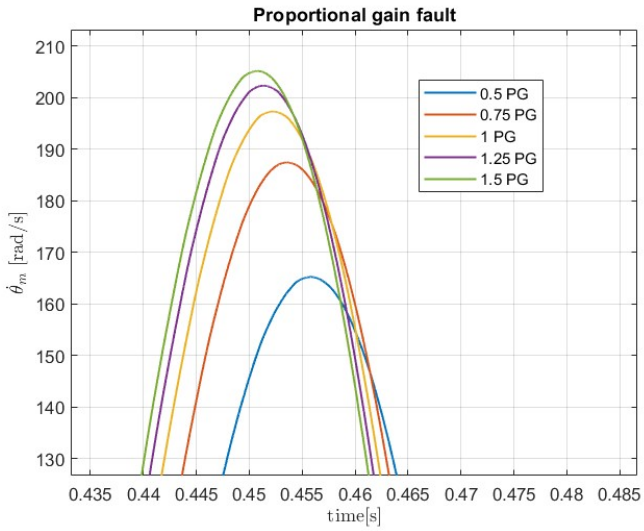

Figure 3.69: Zoom of motor velocity for a chirp command

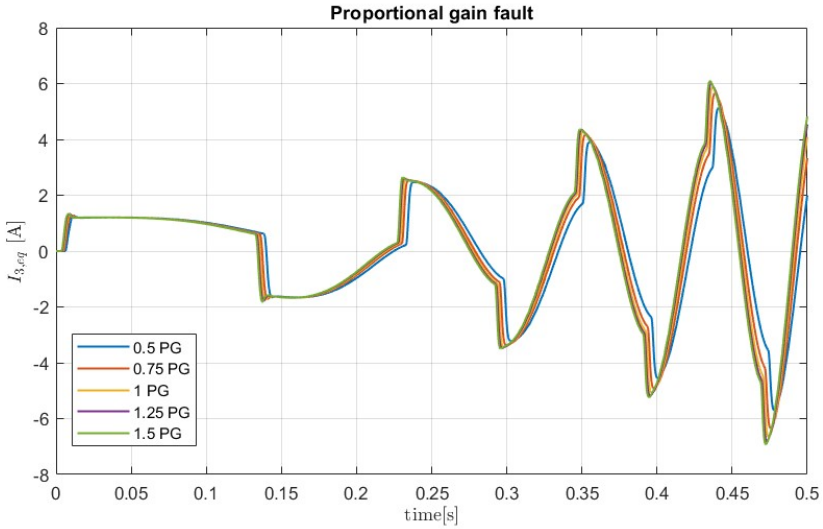

Figure 3.70: Equivalent current for a chirp command

As theory suggests a decrease in proportional gain reduces system readiness. We can notice this phenomenon by observing the speed of the motor in figure 3.68 which fails to reach the nominal conditions. The figure 3.70 confirms this: in changes of direction the current supplied is less than the nominal conditions. The same idea applies to proportional gains exceeding the nominal conditions. In this case, we can expect the system's response to be the opposite of the previous one.

Chapter 4

Bio-inspired algorithms

4.1 Overview

Bio-inspired optimization algorithms are now used on a large scale to solve optimization problems. There are a large number of them in the literature and each has its own peculiarities and limitations. To get an idea of how the scientific community has moved more and more interest in these optimization methods, let's see in the figure 4.1 the number of publications in the period 2005-2019 that contain bio-inspired optimization and nature-inspired optimization in the title, abstract and/or keywords.

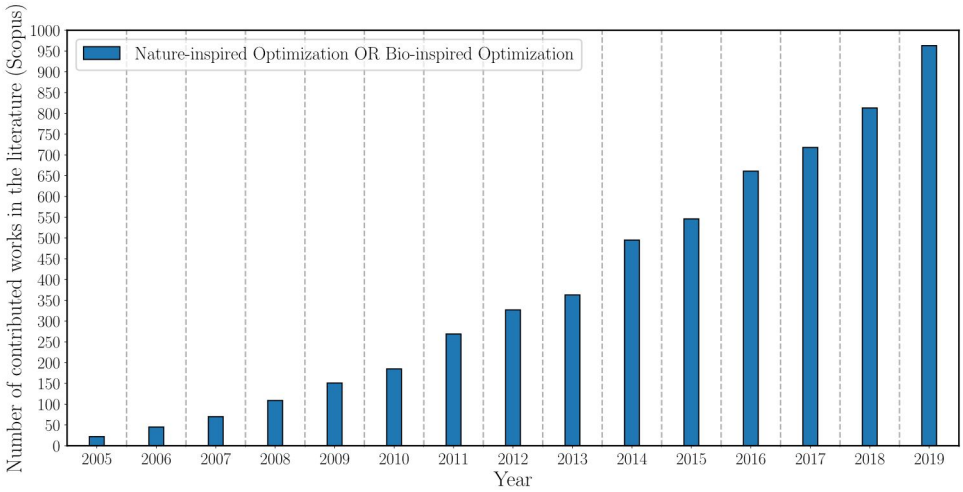

Figure 4.1: Number of papers about bioinspired/nature inspired algorithm [31]

Optimization problems aim, through multiple iterations, to find a solution that is the best among all those considered [32][33][15][34][35]. They come in various

forms in all fields such as aircraft and missile design, production engineering, drug design, finance, health care etc...

As summarized in figure 4.2, optimization algorithms are divided into deterministic and stochastic. The former, already present in the literature for many years, propose exact solutions but have some limitations. The need to make many initial assumptions, the ever-increasing size of the problem due to the large number of available data and the high non-linearity make these algorithms ineffective. Indeed, they would require an excessive computational effort. Greater flexibility and easier testing has been achieved by developing new stochastic algorithms capable of handling large optimization problems [32][33][15][34][35].

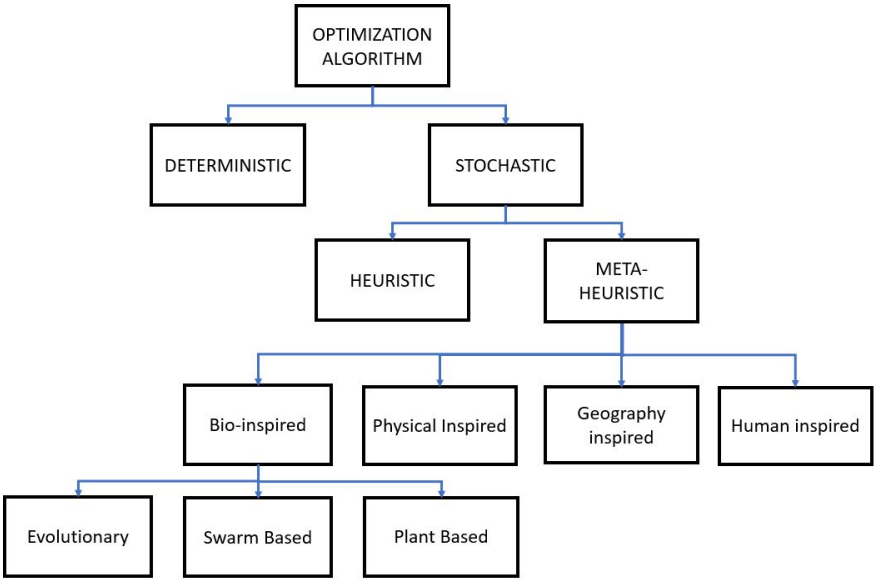

Figure 4.2: Families of optimization algorithms[33]

The term meta and heuristic in Greek mean "high level" and "to find" respectively. In fact, metaheuristic algorithms enhance the search characteristics of heuristic algorithms. The latter have a good efficiency in terms of computational cost but without guaranteeing an optimal solution. Metaheuristic are inspired by biological behavior species such as, for example, animals or groups of insects, to physical phenomena or even to geographical and human phenomena. So bio-inspired algorithms are just one family of all metaheuristic. They can be further divided into Evolutionary Algorithm (EA), Swarm based algorithm (SBA) and Plant based algorithm (PB)[32][33]. The evolutionary algorithm take inspiration from nature and are based on the Darwinian theory of evolution. The initial population evolves at each iteration towards a better condition through processes of recombination, mutation and selection. The individuals who will present the

best fitness function (the most fit to survive) will be able to reproduce more and go towards better research points based on the information provided by the research space (environment) [36]. We can mention the two most famous algorithm: the genetic algorithm (GA) and the differential evolution (DE). Swarm based methods are based on the behavior of swarms and groups of biological beings. The interaction between them and the exchange of information and resources (carried out voluntarily or not) is itself a form of optimization towards a better general condition of the swarm, not obtainable by the single individual. These links can be described by mathematical formulations and consequently applied to optimization problems of our interest. This exchange of information and feedback between individuals in the population represents the strength of swarm based and the major difference with EAs. The most famous and used swarm based is certainly the Particle Swarm Optimization (PSO).

For the sake of brevity, we report below a partial list of some of the most famous algorithms used in the literature. Referring to [31], we can instead have a huge collection of even lesser-known algorithms.

- Genetic Algorithm (GA), 1992;
- Particle Swarm Optimization (PSO), 1995;
- Ant colony Optimization (ACO), 1996;
- Differential Evolution (DE), 1997;
- Artificial Bee Colony (ABC), 2005;
- Gravitational Search Algorithm (GSA), 2009;
- Grey Wolf Optimization (GWO), 2014;
- Crow Search Algorithm (CSA), 2016;
- Whale Optimization Algorithm (WOA), 2016;
- Grasshopper Optimisation Algorithm (GOA), 2017;

As anticipated before, each algorithm proposes a different approach to resolution and therefore some will be more suitable to be applied to certain problems than others. For example ACO and PSO have strong global search capabilities but the former has a slow search speed while the latter is prone to premature convergence[37].

This is how it becomes important to take into account, when applying a certain method, characteristics such as convergence speed, predisposition to avoid local minima (or maxima), exploration and exploitation capability. Exploration phase

evaluates all the available search space going to identify the set of solutions close to the optimum, while exploitation focuses on finding optimal solutions among the best candidates chosen in the previous phase[38]. In the beginning, more exploration helps avoid local optima, while later, more exploitation refines the solution. Balancing exploration and exploitation is key for an efficient search process.

To overcome at least in part the characteristic limitations of some algorithms and improve performance in the search for the optimal solution, it is possible to resort to the combination of several algorithms by creating hybrids. An example can be the hybridization of two of the most famous: the PSO and the GWO. Their union allows to obtain the good exploration capabilities of the GWO and good exploitation capabilities of the PSO [39].

4.2 Chosen algorithms

Among the numerous algorithms present in the literature, the choice of which to use for this work was made taking several aspects into consideration. The idea is to explore the potential of methodologies that are as recent as possible compared to more traditional techniques that have already been tested, without however neglecting the performance obtained in previous applications. The choice fell on the following three algorithms: the Sparrow Search Algorithm (SSA), the Honey Badger Algorithm (HBA) and the Dandelion Optimizer Algorithm (DOA). Not being able to refer to applications relating to prognostic problems for the failures of electromechanical actuators, the three algorithms were also chosen thanks to the support material present on the Mathworks portal.

4.2.1 Sparrow Search Algorithm (SSA) in theory

Proposed by Jiankai Xue and Bo Shen in 2020, Sparrow search is part of the swarm intelligence family and one of the most recent and robust algorithms still available. In fact, as reported in [37][38], it presents superior performances to the more classic PSO and GWO after the results obtained through the test functions. We will use the original version of the algorithm proposed in [37], but keep in mind that numerous other reworks have been developed to mitigate the premature convergence that occasionally occurs[40]. The algorithm is inspired by the behavior of flocks of birds in obtaining food[37]. They have superior intelligence to other bird species and maintain patterns and a well-established hierarchy over time. In particular, the population is divided into hunters or producer and scrounger each with well-defined tasks. The former have the task of searching for food while the latter will obtain food directly from the former. Producers can become scrounger and vice versa at any time when an individual's energy level (fitness function)

exceeds a certain level. Everyone is constantly monitoring each other's behavior. Furthermore, sparrows tend to constantly seek a better position within the group to avoid exposing themselves too much to predators. To build the mathematical model we summarize the behavior of pigeons in the following points:

- The energy level, provided by the fitness function, determines the individuals in charge of looking for food but the proportion between producers and consumers must be constant. Scroungers try to raise their energy level to become producers[37];
- When a predator is detected, an individual (called scouter) sends out an alarm signal which forces the producers to guide the whole flock towards another search area[37];
- Sparrows at the edges of the flock try to move towards the center to defend themselves from external predators[37];

Referring now to the figure 4.3 obtained from [40] we learn about the sequence of operations implemented by the algorithm to reach the optimal solution.

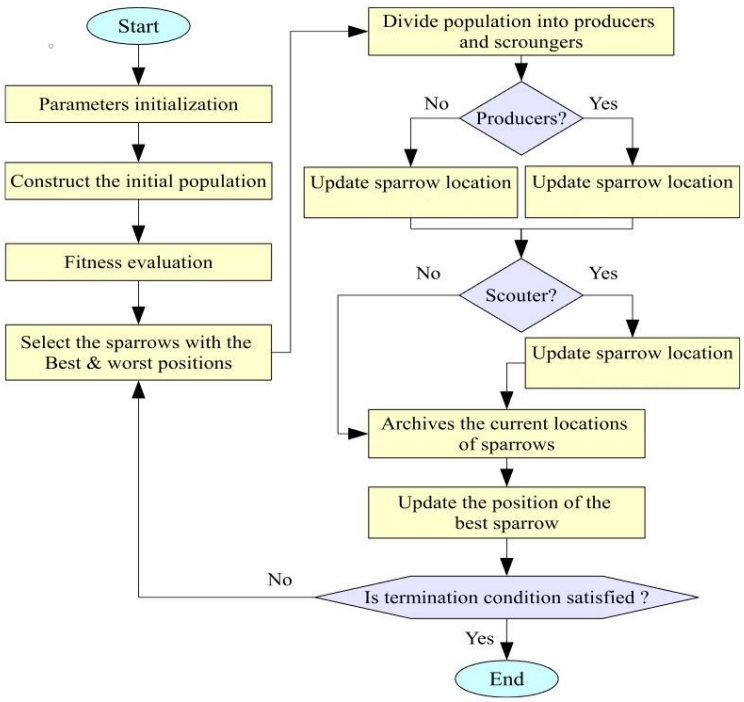

Figure 4.3: Flow-chart of SSA [40]

We therefore identify a first phase of initialization of the parameters such as

for example the initial population and the search space. Reminding us that meta-heuristic algorithms introduce an initial randomization into the search process[37]. Subsequently, the individuals with the best and worst fitness function, respectively, are identified. The population of sparrows and their fitness function can be defined respectively via the 4.1 and 4.2 matrices.

$$X = \begin{bmatrix} x_{1,1} & ... & x_{1,d} \\ ... & ... & ... \\ x_{n,1} & ... & x_{n,d} \end{bmatrix} \tag{4.1}$$

$$F_X = \begin{bmatrix} f([x_{1,1}, ..., x_{1,d}]) \\ ... \\ f([x_{n,1}, ..., x_{n,d}]) \end{bmatrix} \tag{4.2}$$

Where n is the number of individuals, d is the size of the problem (variables to be optimized), each row of 4.1 corresponds to a sparrow and each row of 4.2 corresponds to the fitness function of the associated sparrow. Finally, the real iterative cycle will take place which will update the best solution found so far at each iteration.

4.2.2 Honey Badger Algorithm (HBA) in theory

The second algorithm we will study is the Honey Badger algorithm. Like the SSA, it is part of the swarm intelligence family. Recently developed, it was proposed in its first version by Fatma A. Hashim, Essam H. Houssein, Kashif Hussain, Mai S. Mabrouk, Walid Al-Atabany in 2021 [41]. Although meta-heuristic algorithms were born to manage the complexity of optimization problems in terms of non-linearity and size of the search space, some of them suffer from premature convergence towards local minima (or maxima)[41][42]. The HBA aims to overcome this problem thanks to a good balance between the explotiation and exploration phases. From the analysis carried out in x and y, it appears to outperform the more classic algorithms such as the PSO and the GWO[41][42][43].

The HBA is inspired by the behavior of the honey badger in search of food, a mammal that lives in the semi-desert areas of Africa. It feeds on even large preys that it searches for in the subsoil or on honey. We can distinguish two phases in the supply of food: the first is called digging mode, while the second honey mode. In digging mode the honey badger relies on its excellent nose to locate approximately the location of the prey (exploration) and subsequently chooses the most appropriate point where to make the hole to reach it (exploitation). In honey mode, however, he relies on a bird to guide him directly to the hive[41]. To understand what the main phases of the HBA are, let us refer to the flowchart in figure 4.4.

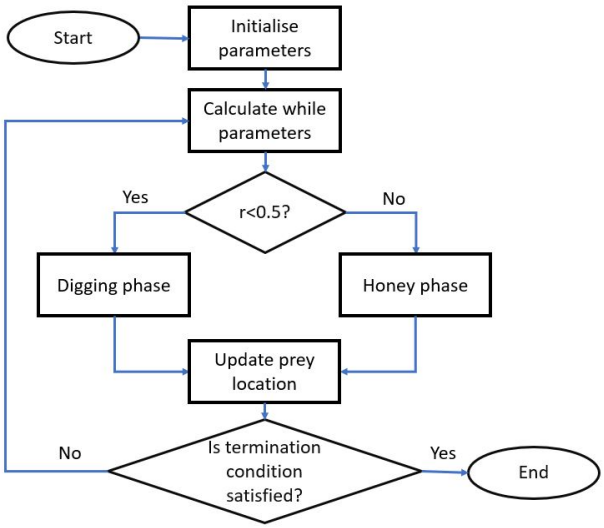

Figure 4.4: Flow-chart of HBA

What we can immediately notice is the close resemblance to the SSA. The first phase of initialization consists in setting the variables used during the iterations of the algorithm such as the stop criteria, the initial population and the first calculation of the fitness function. We define the population and consequently the fitness function in the same way as the SSA using 4.1 and 4.2. We then note the two search phases (digging and honey) chosen on the basis of a random parameter r initialized at each cycle. It will take on a value between 0 and 1.

4.2.3 Dandelion optimizer algorithm (DOA) in theory

The last algorithm we will discuss is the Dandelion optimizer (DOA). Proposed in 2022 by Shijie Zhao[44]. It is one of the most recent methods available in the literature and its basic version is available on [45]. The excellent performances obtained in [46], by applying this methodology to CEC2017 benchmark functions and comparing them with 9 known algorithms, encourage us to apply the algorithm to our case study.

The Dandelion optimizer is based on the reproduction cycle of the dandelion, a plant composed of a stem and a head. The latter has a spherical shape and is made up entirely of seeds which will be transported by the wind to allow continuity of the species. They can move tens of kilometers thanks to the hairs they are equipped with and the vortices that are generated around them[46][44]. The cycle divides into 3 phases: the rising stage, the descending stage and the landing stage. The first is strongly influenced by wind and weather. In fact, in case of rain the seed will only fall near the plant[46][44].

Let us now introduce the Danelion optimizer workflow with figure 4.5 and note a certain similarity in terms of setup with the previous two.

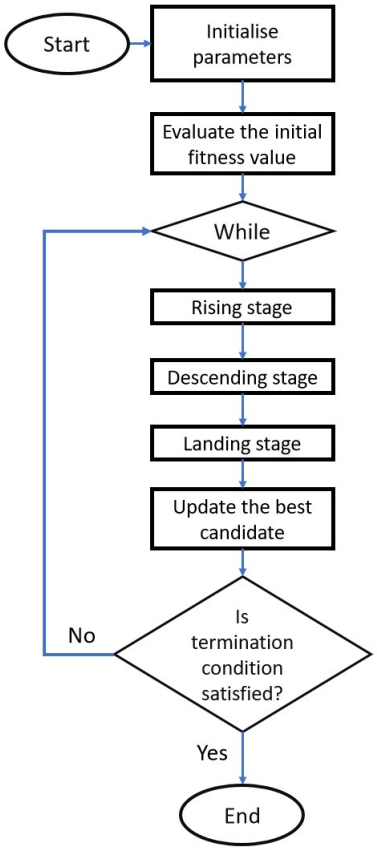

Figure 4.5: Flow-chart of DOA

The first part is characterized by the initialization of the parameters. In particular, in this case it will be necessary to build the initial random population of 50 individuals. We will therefore refer to matrix 4.1. For each individual the fitness function will be calculated and based on the values obtained all individuals will be sorted in ascending order indicating X_{elite} the one with the lowest fitness function. Finally, the iterative cycle, containing the three phases mentioned above, can start by updating the X_{elite} at each iteration.

Chapter 5

Algorithms implementation and results

5.0.1 Fitness function

The optimization algorithms pursue the optimal solution by applying evaluation criteria, specific to each algorithm, at each iteration carried out. The set of these criteria is called *Fitness Function*. In our case, at each iteration, a dynamic response will be generated by simulating the Monitor model. This dynamic response is translated into the equivalent current and is called iLF. It is compared with the counterpart of the Reference model (iHF) which represents the response that the real system would have in the presence of a failure among those described in paragraph 3.5.

To faithfully replicate the behavior of the reference model, the response of the monitor model will be modified by varying the parameters on which it is based. In particular, we will have 8 coefficients k_i, normalized through a linear interpolation, whose value can oscillate between 0 and 1 based on the entity of the failure. The goal is to find the combination of parameters that minimize the difference between the two currents iLF and iHF. Below we see for each failure which range we are going to interpolate:

- Dry friction $[k_1]$: we assume the design conditions as nominal conditions, therefore $k_1 = 0$. While the maximum failure is obtained with friction three times the nominal conditions, therefore $k_1 = 1$;
- Backlash $[k_2]$: the nominal conditions correspond to $k_2 = 0$ while $k_2 = 1$ corresponds to a backlash one hundred times the nominal conditions;
- Short Circuit $[k_3][k_4][k_5]$: respectively for phase A, B and C. If the phase is perfectly functional we will have a coefficient k equal to zero. As the

percentage of windings are short circuited, the variable k grows up to the theoretical value of one. As regards the short circuit we will only evaluate it for one phase.

- Eccentricity $[k_6][k_7]$: coefficients used respectively to indicate the modulus and phase of the eccentricity. As regards the module, the perfect concentricity will have $k_6 = 0$ while in case of contact between stator and rotor $k_6 = 1$. The phase, which influences the operation of the motor only in case of k_6 different from zero, will have a value $k_7 = 0$ if the phase is $-\pi$ while $k_7 = 1$ if the phase is π. The nominal conditions intuitively set $k_7 = 0.5$.

- Proportional gain $[k_8]$: for a 50% reduction of the nominal conditions we will have $k_8 = 0$ while for a 50% increase of the nominal conditions we will have $k_8 = 1$. As in the case of the eccentricity phase, the nominal conditions will be represented by $k_8 = 0.5$.

The fitness function receives as input a vector $\vec{k}$ of 8 components, initially random as foreseen by the metaheuristic algorithms and subsequently at each iteration it will update them according to the instructions provided by the chosen algorithm.

The correspondence between the responses of the two models (the similarity between the two equivalent currents produced) will be evaluated using the root mean square error RMSE (eq. 5.1).

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{5.1}$$

Where n represents the number of measurements over time (in our case the simulation time interval divided by the integration step), y_i and $\hat{y}_i$ represent the i_{th} measurement performed and the i_{th} exact value, respectively[47]. The RMSE, widely used in the scientific world as a parameter in evaluating the performance of different models[47], indicates how well the iLF current vector (the measurement performed) approximates the iHF vector (the exact value).

Each failure will be evaluated for two levels of intensity: low ($k_i = 0.25$) and high ($k_i = 0.75$) and for each case 10 optimizations will be performed. Next, we will analyze the last scenario characterized by multiple low-intensity random failures in order to get closer to actual operating conditions. In fact, it is important to identify potential problems in their early stages of development.

Once the optimization has been completed, the effectiveness of the algorithm used will be evaluated by relating the single coefficient of the vector k_i to the theoretically expected value $k_{i,HF}$ via the equation 5.2.

$$Err[\%] = \frac{100}{\sqrt{6.5}} \cdot \sqrt{\sum_{i=1}^{6}(k_i - k_{i,HF})^2 + k_{6,HF} \cdot (k_7 - k_{7,HF})^2 + (k_8 - k_{8,HF})^2} \quad (5.2)$$

Particular attention is paid to the eccentricity phase coefficient k_7. Its error, in fact, is not taken into consideration if the eccentricity module is in nominal conditions. Furthermore, the maximum error with respect to the nominal conditions is obtained when we obtain all the k_i coefficients equal to 1. In this case the root term will be equal to 6.5. We then normalize the error with respect to this value.

The flowchart in figure 5.1 summarizes the workflow adopted in this paper for failure prognostic purposes.

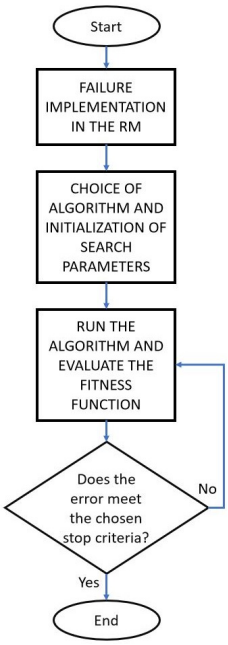

Figure 5.1: Workflow for prognostic purpose

Before continuing, let's spend just a few words regarding the strategies adopted to make the workflow more efficient. The algorithms have been implemented in Matlab in version 2022b, which, as reported by the Mathworks website[48], has more than doubled performance compared to the 2015a version. Numerous test simulations were then performed to identify bottlenecks in the code. In particular, it was found that the line of code $simOut = sim('Monitor', [], options)$, contained in the fitness function, occupies 97% of the total calculation time. On the other hand, the monitor model will be simulated, at each iteration, for all individuals in the population. The monitor model was then set in *accelerator* mode and the

fast restart was activated. The latter keeps the model compiled when multiple simulations need to be carried out in an iterative cycle. Thanks to these measures the simulation time was thus reduced by 87%.

The stopping criteria of the iterative loop in figure 5.1 play a crucial role in defining the computational performance. Based on the calculation times we expect and the degree of precision required, we can opt respectively for a maximum number of iterations or by setting a tolerance in the error of the solution found. In our work we will opt for a mixed solution: the maximum number of iterations that we will impose will be 150 and the RMSE tolerance equal to 10^{-3}. Whenever one of the two values is reached the optimization will be concluded.

5.0.2 Sparrow Search Algorithm (SSA) in practice

Based on what was said in paragraph 4.2.1 we now implement the SSA script, whose pseudocode is shown below.

```
\\ PARAMETER INITIALIZATION
   Set Max_iter and toll (Maximum number of iteration and tolerance);
   Set NP (number of producer);
   Set NS (number of scrounger);
   Set SC (number of scouter);
   Set the initial population and compute the fitness function;
\\ WHILE RMSE > toll && iter < Max_iter
        Rank the fitness values and find the best individual and the worst individual;
        \\ FOR 1:NP
                Update the producer position;
        \\ END FOR
        \\ FOR 1:NP
                Update the scrounger position;
        \\ END FOR
        \\ FOR 1:SC
                Update the scouter position;
        \\ END FOR
Get the current new location and If the new location is better than before, update it;
\\ END WHILE
\\ RETURN Xbest,RMSE,time
```

The first step consists in initializing the necessary values (number of sparrows, number of producers, scroungers and scouters, tolerance and maximum number of iterations). The matrix of the virtual population of sparrows will have dimension [50x8], where 8 is precisely the number of k variables to be optimized while 50 is the number of individuals. Of these 50 individuals, 20% will represent the producers and the rest will assume the function of scrounger. 15% of the total instead (the scouters) will have the task of emitting a signal if a predator is detected. Each value of the matrix will initially assume a random value between 0 and 1. Then, the fitness function of each individual will be evaluated by identifying the best and worst individual. Finally, the iterative while loop will look for the desired optimal solution. As explained in paragraph 5.0.1, to have a good compromise between good results and reasonable calculation time, a tolerance of 10^{-3} and a maximum number of iterations of 150 were assumed. The following tables report the results obtained from the simulations.

Table 5.1: Low Friction F=1.5 - SSA

N° opt	$k_1 = 0.25$	$k_2 = 0$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.2304	0.0016	0.0023	0.0000	0.0006	0.0007	0.1419	0.4930	0.83%
2	0.2369	0.0000	0.0001	0.0000	0.0000	0.0003	0.0034	0.5010	0.52%
3	0.2313	0.0000	0.0017	0.0000	0.0046	0.0064	0.0078	0.4986	0.80%
4	0.2294	0.0000	0.0000	0.0000	0.0000	0.0074	0.0000	0.4951	0.88%
5	0.2357	0.0000	0.0000	0.0000	0.0000	0.0000	0.4711	0.4942	0.61%
6	0.2315	0.0000	0.0000	0.0012	0.0008	0.0018	0.0195	0.4968	0.74%
7	0.2345	0.0034	0.0000	0.0048	0.0000	0.0006	0.6211	0.4958	0.67%
8	0.2362	0.0002	0.0021	0.0010	0.0000	0.0047	0.0000	0.5009	0.58%
9	0.2343	0.0000	0.0018	0.0000	0.0000	0.0013	0.0000	0.5011	0.63%
10	0.2363	0.0000	0.0000	0.0041	0.0000	0.0067	0.1317	0.4975	0.63%

Table 5.2: High Friction F=2.5 - SSA

N° opt	$k_1 = 0.75$	$k_2 = 0$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.6973	0.0013	0.0019	0.0067	0.0030	0.0076	0.3047	0.5003	2.11%
2	0.6972	0.0000	0.0079	0.0028	0.0000	0.0116	0.0867	0.4935	2.16%
3	0.7107	0.0005	0.0000	0.0001	0.0088	0.0000	0.0176	0.4975	1.58%
4	0.6952	0.0000	0.0107	0.0039	0.0000	0.0094	0.0026	0.5024	2.23%
5	0.6978	0.0000	0.0000	0.0030	0.0036	0.0024	0.6698	0.5077	2.08%
6	0.7029	0.0000	0.0000	0.0000	0.0000	0.0019	0.0034	0.5009	1.85%
7	0.7056	0.0000	0.0086	0.0000	0.0000	0.0000	0.0085	0.4970	1.78%
8	0.6872	0.0002	0.0139	0.0000	0.0000	0.0098	0.1606	0.5033	2.55%
9	0.6822	0.0022	0.0100	0.0036	0.0034	0.0000	0.0085	0.5024	2.70%
10	0.6990	0.0000	0.0021	0.0102	0.0046	0.0000	0.0024	0.4946	2.06%

Table 5.3: Low Backlash B=25 - SSA

N° opt	$k_1 = 0$	$k_2 = 0.25$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.0002	0.2212	0.0005	0.0062	0.0001	0.0001	0.0296	0.4841	1.31%
2	0.0001	0.2205	0.0001	0.0056	0.0035	0.0001	0.0001	0.4794	1.43%
3	0.0000	0.2259	0.0002	0.0000	0.0000	0.0000	0.3103	0.4869	1.08%
4	0.0013	0.2233	0.0000	0.0007	0.0069	0.0032	0.0207	0.4889	1.17%
5	0.0000	0.2251	0.0000	0.0000	0.0057	0.0000	0.6362	0.4925	1.04%
6	0.0008	0.2235	0.0008	0.0008	0.0008	0.0008	0.0008	0.4918	1.09%
7	0.0004	0.2258	0.0016	0.0000	0.0001	0.0042	0.0057	0.4957	0.98%
8	0.0036	0.2219	0.0000	0.0014	0.0054	0.0000	0.8553	0.4922	1.17%
9	0.0002	0.2222	0.0002	0.0002	0.0002	0.0011	0.3076	0.4846	1.25%
10	0.0001	0.2245	0.0001	0.0005	0.0002	0.0000	0.9738	0.4912	1.06%

Table 5.4: High Backlash B=75 - SSA

N° opt	$k_1 = 0$	$k_2 = 0.75$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.0003	0.6819	0.0000	0.0283	0.0000	0.0096	0.0271	0.4804	3.02%
2	0.0001	0.7048	0.0000	0.0000	0.0133	0.0000	0.9730	0.4866	1.92%
3	0.0025	0.6976	0.0011	0.0034	0.0002	0.0032	0.0224	0.4831	2.17%
4	0.0000	0.6983	0.0075	0.0018	0.0032	0.0089	1.0000	0.4841	2.17%
5	0.0001	0.7087	0.0000	0.0000	0.0000	0.0035	0.0000	0.4872	1.70%
6	0.0015	0.6867	0.0010	0.0119	0.0000	0.0037	0.5260	0.4761	2.70%
7	0.0021	0.7146	0.0081	0.0000	0.0000	0.0275	0.0747	0.4910	1.82%
8	0.0055	0.7096	0.0000	0.0111	0.0000	0.0072	0.0818	0.4919	1.71%
9	0.0006	0.6960	0.0060	0.0000	0.0000	0.0012	0.8428	0.4811	2.26%
10	0.0000	0.7002	0.0000	0.0157	0.0000	0.0001	0.1244	0.4854	2.13%

Table 5.5: Low Short Circuit N=0.75 - SSA

N° opt	$k_1 = 0$	$k_2 = 0$	$k_3 = 0.25$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.0000	0.0000	0.2249	0.0003	0.0000	0.0037	0.0057	0.4998	0.99%
2	0.0000	0.0000	0.2350	0.0000	0.0000	0.0000	0.0000	0.4911	0.68%
3	0.0001	0.0000	0.2274	0.0000	0.0000	0.0003	0.0000	0.5026	0.89%
4	0.0000	0.0003	0.2232	0.0001	0.0085	0.0111	0.2571	0.5022	1.19%
5	0.0000	0.0000	0.2270	0.0050	0.0000	0.0000	0.1199	0.4931	0.96%
6	0.0022	0.0000	0.2089	0.0022	0.0025	0.0049	0.6215	0.5185	1.78%
7	0.0000	0.0000	0.2265	0.0002	0.0008	0.0032	0.0137	0.4922	0.98%
8	0.0001	0.0000	0.2296	0.0064	0.0000	0.0017	0.1271	0.4923	0.89%
9	0.0039	0.0000	0.2234	0.0003	0.0049	0.0015	0.0070	0.5014	1.08%
10	0.0001	0.0001	0.2296	0.0065	0.0001	0.0001	0.0673	0.4887	0.95%

Table 5.6: High Short Circuit N=0.25 - SSA

N° opt	$k_1 = 0$	$k_2 = 0$	$k_3 = 0.75$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.0000	0.0000	0.7095	0.0215	0.0000	0.0281	0.3453	0.5072	2.13%
2	0.0000	0.0004	0.7197	0.0000	0.0028	0.0001	0.0000	0.5131	1.30%
3	0.0000	0.0000	0.7174	0.0123	0.0000	0.0034	0.0293	0.5059	1.39%
4	0.0153	0.0092	0.6911	0.0299	0.0164	0.0228	0.3295	0.5373	3.25%
5	0.0000	0.0000	0.7183	0.0053	0.0000	0.0034	0.0065	0.5143	1.38%
6	0.0028	0.0007	0.7054	0.0127	0.0191	0.0000	0.2542	0.5117	2.02%
7	0.0029	0.0020	0.7122	0.0039	0.0203	0.0076	0.0000	0.5178	1.86%
8	0.0000	0.0000	0.7134	0.0000	0.0000	0.0000	0.0000	0.5199	1.63%
9	0.0000	0.0000	0.7119	0.0121	0.0101	0.0002	0.3463	0.5131	1.69%
10	0.0093	0.0000	0.7071	0.0048	0.0000	0.0059	0.0072	0.5102	1.79%

Table 5.7: Low eccentricity $\zeta = 0.25$ - SSA

N° opt	$k_1 = 0$	$k_2 = 0$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0.25$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.0000	0.0000	0.0000	0.0000	0.0006	0.2815	0.5041	0.4925	1.27%
2	0.0007	0.0000	0.0000	0.0000	0.0000	0.2700	0.4979	0.4897	0.88%
3	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4841	13.88%
4	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4823	13.88%
5	0.0000	0.0018	0.0000	0.0000	0.0000	0.0000	0.0525	0.4751	13.20%
6	0.0020	0.0020	0.0020	0.0020	0.0021	0.2650	0.4735	0.4995	0.81%
7	0.0000	0.0000	0.0000	0.0009	0.0017	0.2581	0.5013	0.4952	0.38%
8	0.0011	0.0000	0.0017	0.0000	0.0035	0.2697	0.4866	0.4923	0.88%
9	0.0000	0.0000	0.0000	0.0004	0.0000	0.2759	0.4961	0.4936	1.05%
10	0.0000	0.0000	0.0000	0.0000	0.0000	0.2729	0.4965	0.4938	0.93%

Table 5.8: High eccentricity $\zeta = 0.75$ - SSA

N° opt	$k_1 = 0$	$k_2 = 0$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0.75$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.0001	0.0000	0.0000	0.0224	0.0001	0.8652	0.5001	0.4816	4.66%
2	0.0000	0.0000	0.0000	0.0000	0.0000	0.7544	0.5037	0.4546	1.79%
3	0.0011	0.0277	0.0000	0.0000	0.0002	0.0000	0.0039	0.4429	33.99%
4	0.0000	0.0000	0.0083	0.0046	0.0010	0.8088	0.5005	0.4873	2.39%
5	0.0105	0.0000	0.0000	0.0030	0.0000	0.8116	0.4963	0.4999	2.46%
6	0.0047	0.0000	0.0055	0.0042	0.0175	0.8341	0.5014	0.4798	3.48%
7	0.0085	0.0000	0.0000	0.0000	0.0000	0.8058	0.4980	0.4980	2.22%
8	0.0098	0.0000	0.0036	0.0000	0.0001	0.8147	0.5009	0.4963	2.57%
9	0.0000	0.0001	0.0007	0.0000	0.0413	0.8220	0.4966	0.4892	3.28%
10	0.0007	0.0000	0.0046	0.0105	0.0038	0.7534	0.4986	0.4676	1.36%

Table 5.9: Low proportional gain G=0.75 - SSA

N° opt	$k_1 = 0$	$k_2 = 0$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.25$	**Err**
1	0.0001	0.0001	0.0026	0.0019	0.0000	0.0048	0.0000	0.2565	0.34%
2	0.0000	0.0000	0.0013	0.0027	0.0000	0.0005	0.0000	0.2544	0.21%
3	0.0006	0.0000	0.0000	0.0000	0.0050	0.0003	0.0000	0.2556	0.29%
4	0.0004	0.0004	0.0004	0.0004	0.0004	0.0099	0.0004	0.2523	0.40%
5	0.0030	0.0002	0.0000	0.0005	0.0000	0.0000	0.1070	0.2524	0.15%
6	0.0000	0.0000	0.0000	0.0005	0.0003	0.0000	0.0036	0.2537	0.15%
7	0.0020	0.0027	0.0016	0.0016	0.0016	0.0023	0.2560	0.2528	0.22%
8	0.0000	0.0000	0.0000	0.0049	0.0000	0.0025	0.0188	0.2524	0.24%
9	0.0000	0.0003	0.0040	0.0016	0.0000	0.0024	0.0136	0.2550	0.28%
10	0.0000	0.0000	0.0031	0.0017	0.0010	0.0012	0.0066	0.2501	0.15%

Table 5.10: High proportional gain G=1.25 - SSA

N° opt	$k_1 = 0$	$k_2 = 0$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.75$	**Err**
1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0003	0.7326	0.68%
2	0.0001	0.0001	0.0000	0.0000	0.0000	0.0001	0.0003	0.7325	0.69%
3	0.0007	0.0000	0.0002	0.0005	0.0014	0.0007	0.1311	0.7349	0.60%
4	0.0000	0.0000	0.0000	0.0000	0.0000	0.0016	0.0031	0.7317	0.72%
5	0.0001	0.0000	0.0000	0.0001	0.0000	0.0001	0.0000	0.7275	0.88%
6	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.7291	0.82%
7	0.0000	0.0000	0.0001	0.0020	0.0000	0.0002	0.0349	0.7253	0.97%
8	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.7326	0.68%
9	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0164	0.7333	0.65%
10	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.7292	0.82%

As we can see from the results listed above, SSA confirms itself as an excellent optimization method. In fact, the error remains confined well below 3% in almost all cases. We notice a slight increase, of about one percentage point, when for the same failure we move from the low-intensity case to the high-intensity case. Looking again at table 5.7 and table 5.8 we notice four optimizations with an excessive error compared to the general trend. This is due to the fact that the original version of the SSA used in our work, as explained previously, suffers in some cases from premature convergence towards local minimum (or maximum) points. This is why it is important, for the same failure, to carry out multiple optimizations in such a way as to identify solutions that have fallen into the traps of local minima.

We summarize in table 5.11 the performance of SSA in terms of mean error and computational cost. The average calculation time is around 42 minutes. It has been significantly reduced thanks to the improvements seen in section 5.0.1 but not yet negligible if we consider an application of this methodology during the cruise phase of the aircraft.

Table 5.11: Mean performance of SSA

Fault	Mean Err[%]	Computational cost[s]
Low Friction	0.69%	2709
High Friction	2.11%	2663
Low Backlash	1.16%	2158
High Backlash	2.16%	2231
Low Short circuit	1.04%	2247
High Short circuit	1.85%	2451
Low eccentricity	4.72%	2725
High eccentricity	5.82%	2546
Low Gain	0.24%	3061
High Gain	0.75%	2620

We complete the SSA results with the case of multiple low intensity failures. The vector to which we will refer for all three algorithms is the following:

$$[k_1, k_2, k_3, k_4, k_5, k_6, k_7, k_8] = [0.0133,\ 0.05,\ 0.003,\ 0,\ 0,\ 0.012,\ 0.5,\ 0.35] \quad (5.3)$$

As we can see in table 5.12, both the error of approximately 2% and the calculation time of approximately 45 minutes are confirmed in the ranges identified in the single failures.

Table 5.12: Multiple fault - SSA

N° opt	k1	k2	k3	k4	k5	k6	k7	k8	Err
1	0.0083	0.0409	0.0100	0.0000	0.0084	0.0000	0.0068	0.3398	2.29%
2	0.0089	0.0462	0.0008	0.0050	0.0008	0.0054	0.0404	0.3458	2.02%
3	0.0124	0.0454	0.0019	0.0001	0.0003	0.0004	0.0528	0.3484	1.98%
4	0.0087	0.0472	0.0030	0.0133	0.0020	0.0173	0.3572	0.3462	0.87%
5	0.0088	0.0442	0.0099	0.0009	0.0001	0.0001	0.2177	0.3486	1.36%
6	0.0097	0.0444	0.0000	0.0030	0.0026	0.0075	0.0000	0.3476	2.18%
7	0.0121	0.0465	0.0072	0.0015	0.0000	0.0039	0.0976	0.3495	1.77%
8	0.0122	0.0425	0.0027	0.0000	0.0015	0.0000	0.0022	0.3469	2.21%
9	0.0134	0.0476	0.0001	0.0000	0.0001	0.0001	0.2326	0.3527	1.25%
10	0.0088	0.0433	0.0061	0.0030	0.0052	0.0020	0.0690	0.3492	1.94%

5.0.3 Honey Badger Algorithm (HBA) in practice

The Matlab implementation of the honey badger algorithm follows the pseudo-code shown below.

```
\\ PARAMETER INITIALIZATION
Set the initial parameter (Max_iter, toll, Beta, C, population number N);
Rank the fitness values and find the best individual and the worst individual;
\\ WHILE RMSE > toll && iter < Max_iter
        Initialize "I" and "alpha";
        \\ FOR j=1:dim
                \\ IF r<0.5
                Update the position in digging mode;
                \\ ELSE
                Update the position in honey mode;
                \\ END IF
        \\ END FOR
Rank the fitness values and update the prey position;
\\ END WHILE
\\ RETURN Xprey, RMSE, time
```

We have chosen an initial population of 50 individuals as a good compromise between computational efficiency and good coverage of the search space, while β and C are two parameters of the algorithm that significantly affect the results. The best fitness values are achieved with $\beta = 6$ and $C = 2$ as suggested by [41]. I and *alpha*, on the other hand, are initialized at each cycle and are respectively the intensity factor, i.e. the distance from the prey, and the density factor, which allows for a linear transition between the exploration and exploitation phases [41]. As in the previous case, the tolerance imposed on the RMSE will be 10^{-3} and the maximum number of iterations will be 150. We analyze below the results of the optimizations carried out.

Table 5.13: Low Friction F=1.5 - HBA

N° opt	$k_1 = 0.25$	$k_2 = 0$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.2384	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4991	0.46%
2	0.2343	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4959	0.64%
3	0.2343	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4959	0.64%
4	0.2343	0.0000	0.0000	0.0000	0.0000	0.0029	0.0106	0.4959	0.65%
5	0.2343	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.4959	0.64%
6	0.2335	0.0000	0.0000	0.0000	0.0008	0.0029	1.0000	0.5016	0.66%
7	0.2343	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4959	0.64%
8	0.2341	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	0.4954	0.65%
9	0.2343	0.0000	0.0000	0.0000	0.0000	0.0000	0.2149	0.4958	0.64%
10	0.2384	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.4991	0.46%

Table 5.14: High Friction F=2.5 - HBA

N° opt	$k_1 = 0.75$	$k_2 = 0$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.7007	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.5033	1.94%
2	0.7010	0.0000	0.0035	0.0003	0.0000	0.0000	0.0000	0.5025	1.93%
3	0.7036	0.0000	0.0000	0.0000	0.0000	0.0065	0.0000	0.5028	1.84%
4	0.7024	0.0000	0.0000	0.0000	0.0043	0.0040	0.0000	0.5046	1.89%
5	0.7007	0.0000	0.0000	0.0000	0.0000	0.0000	0.9414	0.5024	1.94%
6	0.7006	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	0.5033	1.94%
7	0.7060	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	0.5112	1.78%
8	0.7026	0.0000	0.0000	0.0038	0.0042	0.0000	0.9679	0.5046	1.88%
9	0.7005	0.0000	0.0000	0.0000	0.0000	0.0000	0.9937	0.5041	1.95%
10	0.7008	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.5020	1.93%

Table 5.15: Low Backlash B=25 - HBA

N° opt	$k_1 = 0$	$k_2 = 0.25$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.0015	0.2272	0.0000	0.0000	0.0000	0.0045	0.0903	0.4935	0.95%
2	0.0000	0.2250	0.0000	0.0000	0.0000	0.0000	0.9650	0.4908	1.05%
3	0.0000	0.2251	0.0000	0.0000	0.0000	0.0000	0.0000	0.4922	1.02%
4	0.0000	0.2252	0.0000	0.0000	0.0000	0.0000	1.0000	0.4910	1.04%
5	0.0000	0.2266	0.0009	0.0000	0.0000	0.0000	0.0000	0.4939	0.95%
6	0.0016	0.2269	0.0000	0.0000	0.0001	0.0000	0.9999	0.4932	0.95%
7	0.0000	0.2253	0.0000	0.0000	0.0000	0.0000	1.0000	0.4909	1.03%
8	0.0000	0.2252	0.0000	0.0000	0.0000	0.0000	0.0000	0.4910	1.04%
9	0.0000	0.2261	0.0001	0.0000	0.0000	0.0000	0.9537	0.4937	0.97%
10	0.0000	0.2252	0.0000	0.0000	0.0000	0.0000	0.8838	0.4910	1.04%

Table 5.16: High Backlash B=75 - HBA

N° opt	$k_1 = 0$	$k_2 = 0.75$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.0000	0.7069	0.0000	0.0000	0.0000	0.0000	0.0000	0.4878	1.76%
2	0.0000	0.7078	0.0000	0.0000	0.0000	0.0000	1.0000	0.4878	1.72%
3	0.0000	0.7078	0.0000	0.0000	0.0000	0.0000	1.0000	0.4878	1.72%
4	0.0000	0.7078	0.0000	0.0000	0.0000	0.0000	0.2862	0.4879	1.72%
5	0.0000	0.7087	0.0000	0.0000	0.0000	0.0000	0.0001	0.4880	1.68%
6	0.0000	0.7078	0.0000	0.0000	0.0000	0.0000	0.0194	0.4878	1.72%
7	0.0000	0.7078	0.0000	0.0000	0.0000	0.0000	0.5020	0.4878	1.72%
8	0.0000	0.7069	0.0000	0.0000	0.0000	0.0000	0.0000	0.4878	1.76%
9	0.0000	0.7076	0.0002	0.0000	0.0000	0.0000	0.0436	0.4879	1.73%
10	0.0000	0.7069	0.0000	0.0000	0.0000	0.0000	0.0000	0.4878	1.76%

Table 5.17: Low short circuit N=0.75 - HBA

N° opt	$k_1 = 0$	$k_2 = 0$	$k_3 = 0.25$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.0031	0.0000	0.2297	0.0008	0.0008	0.0072	0.3372	0.5052	0.88%
2	0.0025	0.0000	0.2300	0.0000	0.0000	0.0000	0.0000	0.4965	0.80%
3	0.0000	0.0000	0.2290	0.0000	0.0000	0.0000	1.0000	0.4935	0.86%
4	0.0000	0.0000	0.2324	0.0000	0.0000	0.0000	1.0000	0.4904	0.79%
5	0.0000	0.0003	0.2344	0.0000	0.0017	0.0069	0.9767	0.4974	0.68%
6	0.0000	0.0000	0.2252	0.0000	0.0000	0.0000	0.0000	0.4965	0.98%
7	0.0000	0.0000	0.0000	0.0000	0.0000	0.1235	0.0000	0.5272	10.99%
8	0.0000	0.0000	0.2271	0.0000	0.0027	0.0000	0.9478	0.4968	0.91%
9	0.0000	0.0000	0.2252	0.0000	0.0000	0.0000	1.0000	0.4965	0.98%
10	0.0000	0.0000	0.2285	0.0000	0.0000	0.0000	0.0000	0.4936	0.88%

Table 5.18: High short circuit N=0.25 - HBA

N° opt	$k_1 = 0$	$k_2 = 0$	$k_3 = 0.75$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.0000	0.0000	0.7223	0.0000	0.0000	0.0043	0.5390	0.5077	1.14%
2	0.0000	0.0000	0.7218	0.0000	0.0000	0.0000	1.0000	0.5110	1.19%
3	0.0000	0.0000	0.7181	0.0000	0.0000	0.0000	1.0000	0.5149	1.38%
4	0.0000	0.0000	0.7213	0.0000	0.0000	0.0000	0.0000	0.5093	1.18%
5	0.0000	0.0000	0.7218	0.0000	0.0000	0.0000	0.0000	0.5094	1.17%
6	0.0027	0.0000	0.7183	0.0011	0.0000	0.0000	0.1079	0.5144	1.37%
7	0.0000	0.0000	0.7183	0.0000	0.0000	0.0000	0.0000	0.5118	1.33%
8	0.0000	0.0000	0.7183	0.0000	0.0000	0.0000	0.0000	0.5118	1.33%
9	0.0000	0.0000	0.7213	0.0000	0.0000	0.0000	0.0000	0.5091	1.18%
10	0.0000	0.0005	0.7187	0.0000	0.0033	0.0000	0.8619	0.5089	1.28%

Table 5.19: Low eccentricity $\zeta = 0.25$ - HBA

N° opt	$k_1 = 0$	$k_2 = 0$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0.25$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.0000	0.0000	0.0000	0.0000	0.0000	0.2723	0.5006	0.4904	0.95%
2	0.0000	0.0000	0.0000	0.0000	0.0000	0.2703	0.5005	0.4932	0.84%
3	0.0000	0.0000	0.0000	0.0000	0.0000	0.2797	0.4967	0.4936	1.19%
4	0.0000	0.0000	0.0000	0.0000	0.0000	0.2706	0.5010	0.4939	0.84%
5	0.0000	0.0000	0.0000	0.0000	0.0000	0.2845	0.5010	0.4957	1.36%
6	0.0000	0.0000	0.0000	0.0000	0.0003	0.2745	0.4967	0.4941	0.99%
7	0.0000	0.0000	0.0000	0.0000	0.0000	0.2813	0.4964	0.4944	1.25%
8	0.0000	0.0000	0.0000	0.0000	0.0000	0.2792	0.4967	0.4934	1.18%
9	0.0000	0.0000	0.0000	0.0000	0.0000	0.2691	0.5012	0.4940	0.78%
10	0.0000	0.0000	0.0000	0.0000	0.0000	0.2704	0.5003	0.4932	0.84%

Table 5.20: High eccentricity $\zeta = 0.75$ - HBA

N° opt	$k_1 = 0$	$k_2 = 0$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0.75$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.0000	0.0000	0.0000	0.0000	0.0000	0.8323	0.4992	0.4938	3.24%
2	0.0000	0.0000	0.0001	0.0000	0.0000	0.8338	0.4992	0.4941	3.30%
3	0.0000	0.0000	0.0000	0.0000	0.0000	0.8310	0.4995	0.4885	3.21%
4	0.0000	0.0000	0.0000	0.0000	0.0000	0.8331	0.4992	0.4942	3.27%
5	0.0000	0.0000	0.0000	0.0000	0.0000	0.8282	0.4998	0.4957	3.07%
6	0.0026	0.0000	0.0000	0.0001	0.0000	0.8194	0.5000	0.4940	2.74%
7	0.0000	0.0000	0.0000	0.0000	0.0000	0.8338	0.4996	0.4931	3.30%
8	0.0000	0.0000	0.0000	0.0000	0.0000	0.8301	0.4998	0.4945	3.15%
9	0.0000	0.0000	0.0000	0.0000	0.0079	0.8324	0.4998	0.4918	3.26%
10	0.0000	0.0000	0.0000	0.0000	0.0036	0.8254	0.5014	0.4908	2.98%

Table 5.21: Low proportional gain G=0.25 - HBA

N° opt	$k_1 = 0$	$k_2 = 0$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.25$	**Err**
1	0.0007	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.2556	0.22%
2	0.0000	0.0000	0.0000	0.0000	0.0020	0.0000	0.6141	0.2563	0.26%
3	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.2523	0.09%
4	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9990	0.2524	0.09%
5	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.2524	0.09%
6	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.2524	0.09%
7	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.2524	0.09%
8	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.2524	0.09%
9	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.2524	0.09%
10	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1120	0.2524	0.09%

Table 5.22: High proportional gain G=0.75 - HBA

N° opt	$k_1 = 0$	$k_2 = 0$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.75$	**Err**
1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.7326	0.68%
2	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9676	0.7326	0.68%
3	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.7326	0.68%
4	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0801	0.7326	0.68%
5	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.7326	0.68%
6	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1127	0.7326	0.68%
7	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.7326	0.68%
8	0.0009	0.0000	0.0000	0.0000	0.0000	0.0000	0.0306	0.7345	0.61%
9	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.7326	0.68%
10	0.0000	0.0000	0.0000	0.0017	0.0000	0.0000	0.0000	0.7363	0.54%

HBA is confirmed to be an excellent optimization strategy. Observing the errors reported from the 5.13 to 5.22 table, we note that they remain well confined to around 1% with an increase of approximately one percentage point when moving to

a high intensity failure. The algorithm seems to run into local minima significantly less than the previous one. In fact, the only out-of-scale error (around 11%) that we encounter appears to be in table 5.17. Let us now summarize the average performances of the HBA in table 5.23. The average errors reported for each failure confirm what has been said regarding the precision of the algorithm. The average time taken instead stands at 34 minutes.

Table 5.23: Mean performance of HBA

Fault	**Mean Err[%]**	**Computational cost[s]**
Low Friction	0.61%	2039
High Friction	1.90%	2252
Low Backlash	1.00%	1797
High Backlash	1.73%	2058
Low Short circuit	1.88%	2174
High Short circuit	1.25%	2210
Low eccentricity	1.02%	2018
High eccentricity	3.15%	2003
Low Gain	0.12%	2026
High Gain	0.66%	1822

The last case that we submit to the HBA involves the simulation of multiple low intensity failures. We refer again to vector 5.3 previously used for SSA. The generated results are reported in table 5.24. The error fluctuates in a slightly wider range when compared with any single failure. This may be due to the increased complexity of managing multiple failures simultaneously and the danger of local minima. In any case, the error remains within acceptable values and the calculation time used is even lower than the previous average, settling at 29 minutes.

Table 5.24: Multiple fault - HBA

N° opt	**k1**	**k2**	**k3**	**k4**	**k5**	**k6**	**k7**	**k8**	**Err**
1	0.0000	0.0469	0.0000	0.0083	0.0000	0.0000	0.9181	0.3460	1.97%
2	0.0153	0.0000	0.0000	0.0000	0.0014	0.0000	0.9986	0.3562	2.96%
3	0.0000	0.0462	0.0000	0.0000	0.0000	0.0086	0.4493	0.3476	0.62%
4	0.0000	0.0451	0.0108	0.0000	0.0000	0.0000	0.0000	0.3427	2.31%
5	0.0131	0.0469	0.0000	0.0001	0.0000	0.0000	0.9982	0.3496	2.20%
6	0.0000	0.0474	0.0000	0.0000	0.0000	0.0000	0.0093	0.3458	2.23%
7	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.3478	3.00%
8	0.0000	0.0459	0.0000	0.0000	0.0000	0.0000	0.0408	0.3477	2.11%
9	0.0000	0.0471	0.0000	0.0083	0.0000	0.0000	0.0000	0.3460	2.30%
10	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.3478	3.00%

5.0.4 Dandelion optimizer algorithm (DOA) in practice

Before exposing all the results produced by the DOA, we refer to the pseudo-code shown below. We note that the algorithm does not require the initialization of many parameters other than the initial population, which we always chose to be 50 individuals, the tolerance imposed for the RMSE (10^{-3}) and the maximum number of iterations (150).

```
\\ PARAMETER INITIALIZATION
Set the initial parameter (Max_iter, toll, Initial population of dandelion seeds);
Evaluate the fitness function and find the minimum → Xelite and f_best;
\\ WHILE RMSE > toll && iter < Max_iter
        Rising stage:
            \\ IF randn()<1.5
            Update dandelion seeds in clear day conditions;
            \\ ELSE
            Update dandelion seeds in rainy day conditions;
            \\ END IF
        Descending stage: Update dandelion seeds;
        Landing stage: Update dandelion seeds;
\\ IF f(Xelite) < f(Xbest)
Xbest=Xelite and f_best=f(Xelite);
\\ END IF
\\ END WHILE
\\ RETURN Xbest, RMSE, time
```

We now report the results of the optimizations performed in the following tables.

Table 5.25: Low Friction F=1.5 - DOA

N° opt	$k_1 = 0.25$	$k_2 = 0$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.2351	0.0000	0.0000	0.0007	0.0057	0.0138	0.5532	0.4976	0.83%
2	0.2330	0.0002	0.0001	0.0005	0.0030	0.0024	0.2375	0.4972	0.69%
3	0.2337	0.0002	0.0000	0.0000	0.0000	0.0037	0.0216	0.4974	0.66%
4	0.2346	0.0000	0.0000	0.0014	0.0001	0.0045	0.0525	0.4958	0.65%
5	0.2375	0.0000	0.0000	0.0000	0.0000	0.0023	0.2229	0.4999	0.50%
6	0.2333	0.0000	0.0001	0.0002	0.0064	0.0001	0.8682	0.4963	0.72%
7	0.2337	0.0000	0.0001	0.0000	0.0070	0.0045	0.0005	0.4967	0.73%
8	0.2384	0.0000	0.0000	0.0001	0.0000	0.0000	0.0106	0.4991	0.46%
9	0.2341	0.0000	0.0001	0.0000	0.0003	0.0001	0.1976	0.5001	0.62%
10	0.2389	0.0001	0.0000	0.0000	0.0000	0.0036	0.8943	0.4996	0.46%

Table 5.26: High Friction F=2.5 - DOA

N° opt	$k_1 = 0.75$	$k_2 = 0$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.7019	0.0000	0.0016	0.0000	0.0002	0.0031	0.0352	0.5034	1.90%
2	0.7024	0.0000	0.0000	0.0111	0.0004	0.0019	0.0000	0.5053	1.93%
3	0.7018	0.0000	0.0002	0.0054	0.0001	0.0001	0.6527	0.5031	1.91%
4	0.7008	0.0000	0.0001	0.0001	0.0016	0.0006	0.0044	0.5030	1.93%
5	0.7033	0.0000	0.0002	0.0050	0.0002	0.0074	0.1978	0.5037	1.87%
6	0.7005	0.0000	0.0013	0.0035	0.0000	0.0000	0.4385	0.5040	1.96%
7	0.7052	0.0000	0.0000	0.0019	0.0000	0.0145	0.9198	0.5038	1.86%
8	0.7010	0.0000	0.0015	0.0014	0.0007	0.0000	0.3222	0.5033	1.93%
9	0.7004	0.0000	0.0008	0.0115	0.0057	0.0070	0.4411	0.5034	2.03%
10	0.7010	0.0000	0.0000	0.0067	0.0000	0.0000	0.0548	0.5029	1.94%

Table 5.27: Low backlash B=25 - DOA

N° opt	$k_1 = 0$	$k_2 = 0.25$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.0000	0.2257	0.0001	0.0000	0.0036	0.0073	0.5645	0.4928	1.04%
2	0.0000	0.2264	0.0000	0.0006	0.0000	0.0005	0.0084	0.4938	0.96%
3	0.0001	0.2245	0.0000	0.0001	0.0002	0.0000	0.0000	0.4907	1.07%
4	0.0000	0.2239	0.0000	0.0001	0.0000	0.0009	0.6202	0.4901	1.10%
5	0.0002	0.2284	0.0001	0.0007	0.0000	0.0041	0.0458	0.4943	0.89%
6	0.0000	0.2250	0.0000	0.0016	0.0030	0.0120	0.5501	0.4925	1.13%
7	0.0000	0.2250	0.0001	0.0011	0.0012	0.0102	0.0292	0.4903	1.13%
8	0.0000	0.2267	0.0000	0.0000	0.0010	0.0032	0.0131	0.4951	0.94%
9	0.0006	0.2271	0.0000	0.0008	0.0000	0.0013	0.6812	0.4943	0.93%
10	0.0007	0.2278	0.0000	0.0000	0.0003	0.0028	0.0600	0.4955	0.90%

Table 5.28: High backlash B=75 - DOA

N° opt	$k_1 = 0$	$k_2 = 0.75$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.0006	0.7094	0.0005	0.0001	0.0000	0.0130	0.1614	0.4882	1.74%
2	0.0000	0.7078	0.0000	0.0002	0.0001	0.0002	0.9429	0.4878	1.72%
3	0.0000	0.7078	0.0000	0.0000	0.0000	0.0000	0.0166	0.4872	1.73%
4	0.0000	0.7090	0.0000	0.0000	0.0001	0.0000	0.0882	0.4881	1.68%
5	0.0000	0.7103	0.0000	0.0000	0.0000	0.0083	1.0000	0.4894	1.64%
6	0.0000	0.7071	0.0000	0.0000	0.0000	0.0001	0.8820	0.4879	1.75%
7	0.0000	0.7083	0.0004	0.0000	0.0000	0.0000	0.0137	0.4880	1.70%
8	0.0000	0.7112	0.0000	0.0001	0.0010	0.0022	0.0000	0.4897	1.58%
9	0.0000	0.7100	0.0000	0.0000	0.0014	0.0054	0.2272	0.4892	1.64%
10	0.0000	0.7086	0.0000	0.0000	0.0000	0.0111	0.5123	0.4879	1.75%

Table 5.29: Low Short circuit N=0.75 - DOA

N° opt	$k_1 = 0$	$k_2 = 0$	$k_3 = 0.25$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.0000	0.0000	0.2284	0.0011	0.0000	0.0035	0.0003	0.4970	0.87%
2	0.0053	0.0001	0.2247	0.0001	0.0000	0.0013	0.0003	0.5015	1.02%
3	0.0000	0.0000	0.2280	0.0013	0.0037	0.0000	0.0068	0.4944	0.90%
4	0.0053	0.0000	0.2275	0.0039	0.0010	0.0000	0.5939	0.4996	0.92%
5	0.0000	0.0000	0.2262	0.0000	0.0000	0.0000	0.6995	0.5004	0.93%
6	0.0024	0.0000	0.2294	0.0004	0.0000	0.0000	0.0000	0.4963	0.83%
7	0.0008	0.0000	0.2284	0.0077	0.0017	0.0211	0.5010	0.4966	1.23%
8	0.0003	0.0001	0.2269	0.0072	0.0000	0.0001	0.0157	0.4962	0.96%
9	0.0000	0.0000	0.2302	0.0004	0.0037	0.0078	0.1715	0.4944	0.87%
10	0.0028	0.0000	0.2218	0.0079	0.0026	0.0007	0.0099	0.5044	1.17%

Table 5.30: High Short circuit N=0.25 - DOA

N° opt	$k_1 = 0$	$k_2 = 0$	$k_3 = 0.75$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.0031	0.0000	0.7198	0.0000	0.0000	0.0124	0.4132	0.5133	1.39%
2	0.0000	0.0000	0.7126	0.0098	0.0208	0.0511	0.5100	0.5167	2.72%
3	0.0008	0.0000	0.7207	0.0000	0.0000	0.0006	0.0786	0.5121	1.25%
4	0.0000	0.0000	0.7141	0.0256	0.0061	0.0438	0.4538	0.5130	2.50%
5	0.0000	0.0016	0.7206	0.0000	0.0040	0.0152	0.0228	0.5100	1.36%
6	0.0023	0.0001	0.7192	0.0000	0.0069	0.0020	0.6508	0.5127	1.34%
7	0.0000	0.0020	0.7201	0.0013	0.0000	0.0051	0.0597	0.5109	1.27%
8	0.0008	0.0001	0.7193	0.0000	0.0000	0.0088	0.0278	0.5094	1.31%
9	0.0008	0.0000	0.7197	0.0000	0.0011	0.0001	0.0710	0.5139	1.31%
10	0.0006	0.0000	0.7200	0.0000	0.0000	0.0000	0.9960	0.5107	1.25%

Table 5.31: Low Eccentricity $\zeta = 0.25$ - DOA

N° opt	$k_1 = 0$	$k_2 = 0$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0.25$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.0000	0.0000	0.0000	0.0000	0.0000	0.2704	0.5007	0.4906	0.88%
2	0.0011	0.0000	0.0000	0.0000	0.0000	0.2716	0.5004	0.4941	0.88%
3	0.0013	0.0000	0.0004	0.0018	0.0011	0.2715	0.4988	0.4939	0.88%
4	0.0000	0.0000	0.0000	0.0001	0.0000	0.2676	0.5012	0.4934	0.74%
5	0.0000	0.0000	0.0001	0.0015	0.0041	0.2752	0.5049	0.4915	1.06%
6	0.0000	0.0000	0.0014	0.0010	0.0005	0.2724	0.4967	0.4910	0.95%
7	0.0000	0.0000	0.0000	0.0000	0.0025	0.2792	0.5007	0.4927	1.18%
8	0.0000	0.0000	0.0000	0.0049	0.0011	0.2850	0.5005	0.4906	1.44%
9	0.0000	0.0000	0.0000	0.0000	0.0034	0.2705	0.5004	0.4919	0.88%
10	0.0000	0.0000	0.0000	0.0032	0.0055	0.2903	0.5012	0.4925	1.63%

Table 5.32: High Eccentricity $\zeta = 0.75$ - DOA

N° opt	$k_1 = 0$	$k_2 = 0$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0.75$	$k_7 = 0.5$	$k_8 = 0.5$	**Err**
1	0.0000	0.0000	0.0000	0.0000	0.0000	0.8327	0.4992	0.4941	3.25%
2	0.0011	0.0000	0.0000	0.0000	0.0001	0.8258	0.5004	0.4930	2.99%
3	0.0000	0.0000	0.0000	0.0006	0.0016	0.8371	0.4997	0.4890	3.45%
4	0.0006	0.0000	0.0000	0.0019	0.0003	0.8230	0.4996	0.4929	2.88%
5	0.0001	0.0000	0.0003	0.0000	0.0055	0.8277	0.4992	0.4929	3.07%
6	0.0025	0.0000	0.0003	0.0103	0.0009	0.8220	0.4985	0.4941	2.87%
7	0.0062	0.0004	0.0000	0.0003	0.0000	0.8263	0.4999	0.4944	3.01%
8	0.0000	0.0000	0.0000	0.0000	0.0001	0.8242	0.4990	0.4943	2.92%
9	0.0000	0.0000	0.0000	0.0044	0.0003	0.8241	0.4997	0.4918	2.93%
10	0.0041	0.0000	0.0000	0.0044	0.0001	0.8154	0.4990	0.4919	2.59%

Table 5.33: Low proportional gain G=0.75 - DOA

N° opt	$k_1 = 0$	$k_2 = 0$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.25$	**Err**
1	0.0006	0.0003	0.0000	0.0003	0.0006	0.0015	0.2717	0.2557	0.23%
2	0.0001	0.0010	0.0000	0.0023	0.0000	0.0000	0.0006	0.2561	0.26%
3	0.0010	0.0011	0.0001	0.0096	0.0000	0.0017	0.2766	0.2533	0.41%
4	0.0000	0.0002	0.0000	0.0003	0.0062	0.0009	0.0002	0.2543	0.30%
5	0.0000	0.0000	0.0000	0.0008	0.0012	0.0000	0.0780	0.2564	0.26%
6	0.0000	0.0010	0.0000	0.0008	0.0000	0.0000	0.0020	0.2534	0.14%
7	0.0000	0.0000	0.0005	0.0051	0.0000	0.0036	0.0893	0.2519	0.26%
8	0.0000	0.0000	0.0000	0.0001	0.0055	0.0059	0.0043	0.2537	0.35%
9	0.0007	0.0001	0.0000	0.0003	0.0004	0.0000	0.0122	0.2553	0.21%
10	0.0000	0.0000	0.0019	0.0000	0.0000	0.0001	0.0470	0.2563	0.26%

Table 5.34: High proportional gain G=1.25 - DOA

N° opt	$k_1 = 0$	$k_2 = 0$	$k_3 = 0$	$k_4 = 0$	$k_5 = 0$	$k_6 = 0$	$k_7 = 0.5$	$k_8 = 0.75$	**Err**
1	0.0006	0.0000	0.0000	0.0002	0.0003	0.0006	0.0001	0.7318	0.72%
2	0.0007	0.0000	0.0000	0.0008	0.0000	0.0000	0.8722	0.7315	0.73%
3	0.0010	0.0000	0.0000	0.0003	0.0002	0.0000	0.0127	0.7326	0.68%
4	0.0003	0.0000	0.0000	0.0000	0.0001	0.0000	0.7435	0.7338	0.64%
5	0.0001	0.0000	0.0000	0.0002	0.0000	0.0001	0.0608	0.7326	0.68%
6	0.0005	0.0000	0.0000	0.0000	0.0001	0.0000	0.0389	0.7317	0.72%
7	0.0000	0.0000	0.0001	0.0026	0.0003	0.0003	0.6505	0.7359	0.56%
8	0.0020	0.0000	0.0000	0.0001	0.0000	0.0034	0.0629	0.7316	0.74%
9	0.0005	0.0000	0.0000	0.0000	0.0000	0.0002	0.2340	0.7316	0.72%
10	0.0005	0.0000	0.0000	0.0000	0.0004	0.0000	0.8136	0.7317	0.72%

The error generally remains around 1%, always experiencing a slight increase for the same failure in the transition to the high intensity case. Furthermore, compared to the previous two, it seems that the DOA has a stronger ability to not fall into

local minima. Below we report the average error for each failure and the time taken to reach the optimal solution. The average time taken is exactly the same as the HBA, i.e. 34 minutes.

Table 5.35: Mean performance of DOA

Fault	**Mean Err[%]**	**Computational cost[s]**
Low Friction	0.63%	1914
High Friction	1.93%	1899
Low Backlash	1.01%	1844
High Backlash	1.69%	2247
Low Short circuit	0.97%	2127
High Short circuit	1.57%	1981
Low eccentricity	1.05%	2015
High eccentricity	2.99%	2119
Low Gain	0.27%	1863
High Gain	0.69%	2085

We conclude the analyzes carried out with the DOA by taking into consideration the case of multiple failures. Using vector 5.3 as input we obtain the following results.

Table 5.36: Multiple fault - DOA

N° opt	**k1**	**k2**	**k3**	**k4**	**k5**	**k6**	**k7**	**k8**	**Err**
1	0.0124	0.0457	0.0000	0.0001	0.0000	0.0010	0.0702	0.3480	1.91%
2	0.0132	0.0464	0.0000	0.0000	0.0000	0.0000	0.2000	0.3496	1.38%
3	0.0118	0.0460	0.0002	0.0012	0.0008	0.0132	0.4394	0.3507	0.34%
4	0.0124	0.0457	0.0006	0.0000	0.0000	0.0000	0.0044	0.3480	2.19%
5	0.0141	0.0462	0.0000	0.0000	0.0000	0.0024	0.0007	0.3504	2.19%
6	0.0131	0.0463	0.0000	0.0000	0.0000	0.0004	0.0000	0.3496	2.20%
7	0.0132	0.0464	0.0000	0.0002	0.0000	0.0000	0.0172	0.3496	2.13%
8	0.0133	0.0458	0.0000	0.0001	0.0000	0.0003	0.3855	0.3498	0.70%
9	0.0116	0.0449	0.0005	0.0198	0.0027	0.0376	0.4500	0.3481	1.32%
10	0.0128	0.0465	0.0008	0.0028	0.0000	0.0012	0.0075	0.3507	2.17%

5.0.5 Comparison between algorithms

In this paragraph, based on what has been reported previously, we want to discuss and analyze the results obtained in the optimizations. The errors obtained and the calculation times used for each failure will be compared, in terms of arithmetic average. They will also be related via an index that we will call performance coefficient (PC) to establish which is the best algorithm for our application.

Let's start by comparing the average percentage error with table 5.37 and the respective graph in figure 5.2.

Table 5.37: Mean percentage error

Algorithm	F	B	N	ζ	G	**Mean error**	**Std dev.**
SSA - Low	0.69%	1.16%	1.04%	4.72%	0.24%	1.57%	1.61%
SSA - High	2.11%	2.16%	1.85%	5.82%	0.75%	2.54%	1.72%
HBA - Low	0.61%	1.00%	1.88%	1.02%	0.12%	0.93%	0.58%
HBA - High	1.90%	1.73%	1.25%	3.15%	0.66%	1.74%	0.83%
DOA - Low	0.63%	1.01%	0.97%	1.05%	0.27%	0.79%	0.30%
DOA - High	1.93%	1.69%	1.57%	2.99%	0.69%	1.77%	0.74%
Average	1.31%	1.46%	1.43%	3.13%	0.46%		

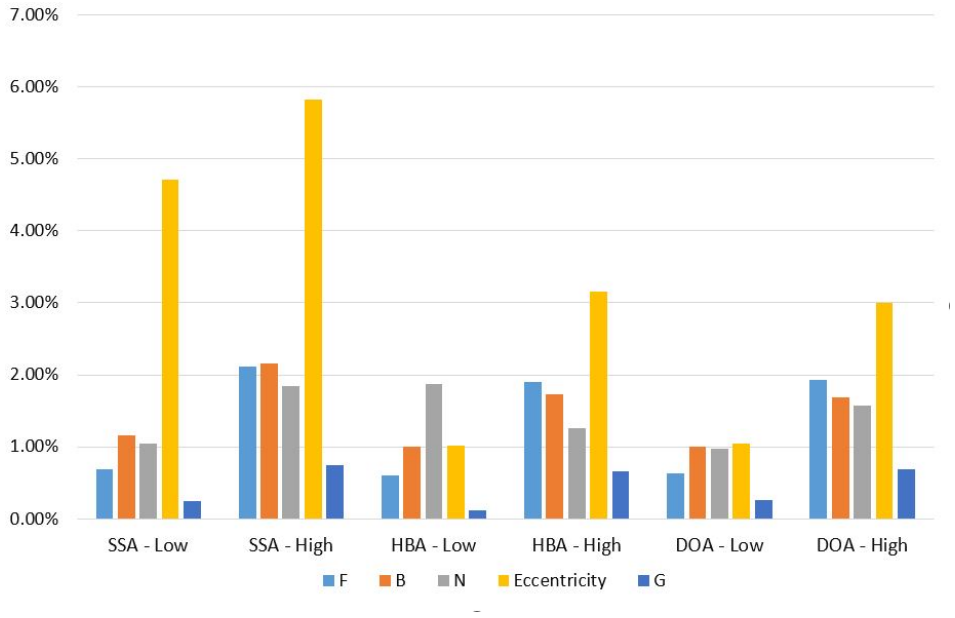

Figure 5.2: Mean percentage error

What we can notice is that the errors remain largely contained below the threshold of 2% with the exception of eccentricity. It, particularly in the case of SSA, presents off-scale error values. This fact is confirmed by referring to the last row (Average) of table 5.37. We also confirm what was anticipated in the previous paragraphs, i.e. a general increase in the error referring to the same algorithm in the transition from the case of low to high severity of a failure while the greatest levels of precision are found for all algorithms regarding the proportional gain. We want to deal with the accuracy of the algorithms also from another point of view, namely the dispersion of the error values through the *standard deviation* reported in the last column of table 5.37. The SSA still suffers the most while the other two algorithms report values similar to each other and about one percentage point lower. Let us remember that the standard deviation tells us how much the

population deviates from its arithmetic mean, indicating the DOA as better from this point of view.

We continue our analysis by evaluating the computational cost using the same strategy used for the average error. Below we report the table and the associated graph.

Table 5.38: Mean computational cost

Algorithm	F[s]	B[s]	N[s]	ζ[s]	G[s]	**Mean total time[s]**	**Std dev.[s]**
SSA - Low	2709	2158	2247	2725	3061	2580.0	334.1
SSA - High	2663	2231	2451	2546	2620	2502.2	153.5
HBA - Low	2039	1797	2174	2018	2026	2010.8	121.2
HBA - High	2252	2058	2210	2003	1822	2069.0	154.2
DOA - Low	1914	1844	2127	2015	1863	1952.6	105.5
DOA - High	1899	2247	1981	2119	2085	2066.2	119.2
Average	2246	2056	2198	2238	2246		

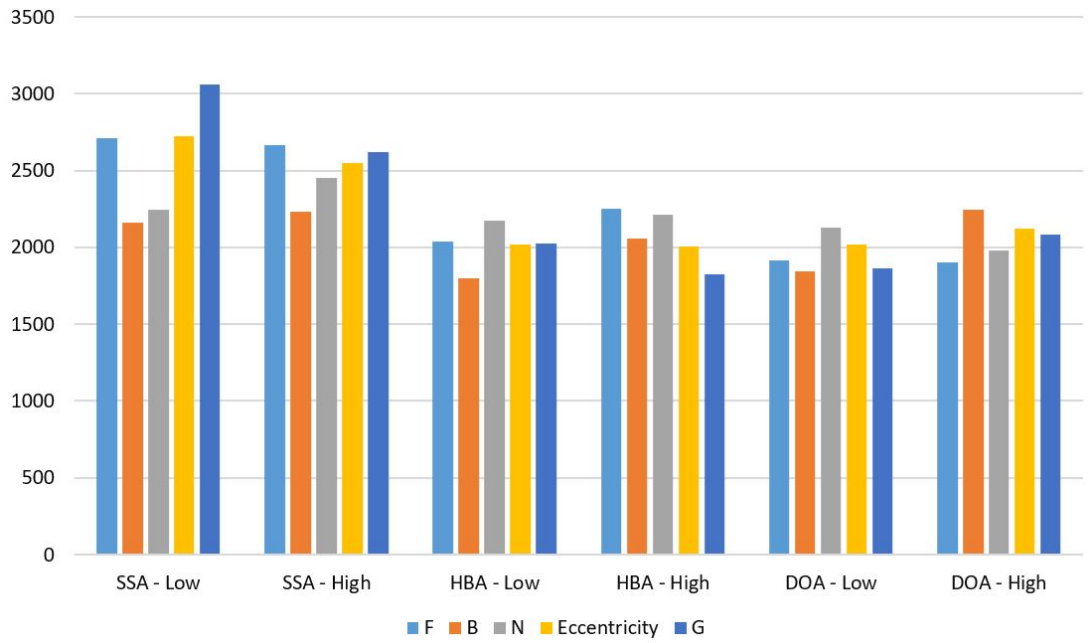

Figure 5.3: Mean computational cost[s]

Looking at the graph in figure 5.3, the SSA still suffers the most also in terms of calculation time, taking approximately 9 minutes more than the other two. Referring to the standard deviation, the HBA and DOA also enjoy a smaller timing discrepancy and are more precise.

The computational cost depends on many factors such as, for example, the

complexity of the algorithm in terms of operations, the precision required, the number of iterations expected and the performance of the machine on which the calculation is performed. For all three algorithms we imposed a minimum tolerance of the RMSE equal to 10^{-3} and a maximum number of iterations equal to 150. In no case was the tolerance reached so this means that all the optimizations performed the 150 expected iterations. Having this similarity between the algorithms, we report below in figure 5.4, as an example, the comparison of the convergence curves in the case of the implementation of multiple failures. Clearly each optimization carried out by the same algorithm will present a slightly different convergence curve in the first iterations caused by the fact that the population is initialized randomly.

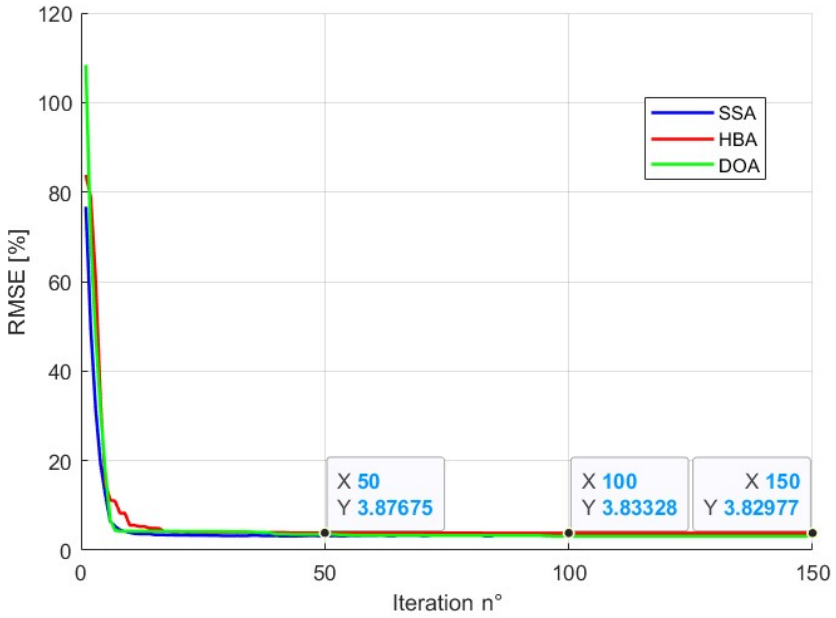

Figure 5.4: Convergence curve

We can see that already from the $50th$ iteration the RMSE is close to its convergence value. The imposed iterations could therefore be greatly reduced, bringing them to the right compromise between precision obtained and time saved. This value could be 100 for example.

We now introduce the *performance coefficient* (PC). It will allow us to link together the error encountered with the calculation time used for each failure and to decide which of the three algorithms is best suited for our work. Based on what is reported in [3] and [15], we define the PC in equation 5.4.

$$PC[\%] = 100 \cdot \left(1 - \frac{Err_i \cdot t_i}{\sum_{i=1}^{3} Err_i \cdot t_i}\right) \tag{5.4}$$

Where Err_i is the percentage error of the failure analyzed and t_i is the time taken expressed in seconds. The denominator has the function of normalizing the value in the numerator with respect to the results of all three algorithms. It is therefore clear how high Err_i and/or t_i values negatively affect the PC. In tables 5.39, 5.40, and 5.41 we calculate the PC for each fault analyzed only after averaging the error and computational cost values between the low and high intensity cases.

Table 5.39: Performance coefficient for SSA

SSA - Fault	**Computational cost[s]**	**Err[%]**	**PC[%]**
Friction	2686	1.40%	57.72%
Backlash	2194.5	1.66%	59.65%
Short circuit	2349	1.44%	64.05%
Eccentricity	2635.5	5.27%	37.63%
Proportional Gain	2840.5	0.50%	54.64%

Table 5.40: Performance coefficient for HBA

HBA - Fault	**Computational cost[s]**	**Err[%]**	**PC[%]**
Friction	2145.5	1.25%	69.72%
Backlash	1927.5	1.37%	70.89%
Short circuit	2192	1.56%	63.62%
Eccentricity	2010.5	2.09%	81.15%
Proportional Gain	1924	0.39%	75.73%

Table 5.41: Performance coefficient for DOA

DOA - Fault	**Computational cost[s]**	**Err[%]**	**PC[%]**
Friction	1906.5	1.28%	72.55%
Backlash	2045.5	1.35%	69.46%
Short circuit	2054	1.27%	72.33%
Eccentricity	2067	2.02%	81.22%
Proportional Gain	1974	0.48%	69.63%

Finally, the average values for each algorithm are reported in Table 5.42. The DOA, not too far from the HBA, is confirmed to be the best candidate offering the best performance. SSA, on the other hand, seems to suffer from a greater calculation time required and a tendency to fall into local minima more easily than the other two. The greater computational time of the SSA is due to the fact that

Table 5.42: Mean final value

Algorithm	**Computational time [s]**	**Err [%]**	**PC [%]**
SSA	2541.1	2.05%	54.74%
HBA	2039.9	1.33%	72.22%
DOA	2009.4	1.28%	73.04%

in the iterative cycle it calls the fitness function 65 times unlike the other two in which it is called only 50.

For completeness of the work, we also report the PC regarding the case of multiple failures. The results collected in table 5.43 confirm what was said above, electing DOA as the most efficient optimization method. The HBA, while maintaining second place, tends to underperform compared to the case of single failures.

Table 5.43: Performance coefficient of multiple fault

Algorithm	**Computational time [s]**	**Err [%]**	**PC [%]**
SSA	2845	1.79%	57.43%
HBA	1743	2.27%	66.93%
DOA	1762	1.65%	75.63%

Chapter 6

Conclusions and future works

To solve the optimization problem, with the aim of predicting failures in electromechanical actuators for primary flight controls, three of the most recent algorithms available were used: the Sparrow search algorithm, the Honey badger algorithm and the Dandelion optimization algorithm.

Based on the results obtained from the simulations, reworked in the form of performance coefficients, we can conclude that DOA is the best candidate for our application both in terms of error and computational cost. The HBA presents very similar, almost identical, values to the previous one, also confirming itself as an excellent optimization strategy. The SSA suffers the most, reporting error values and calculation times higher than the first two.

It is difficult to establish all the causes that lead one algorithm to provide better results than another, but by analyzing their nature, their implementation and the results provided we can certainly draw some considerations. The SSA, compared to the other two, has lower exploration capabilities, in fact, it falls more easily into the trap of local minima. Furthermore, during the parameter initialization phase, the method requires the definition of a greater number of them such as, for example, the percentage of producers, scroungers and other random values useful for the iterative cycle. Since these algorithms are stochastic, small variations in the boundary conditions can vary the solution obtained.

We have seen how the calculation time is strongly influenced by the number of times the fitness function is calculated and therefore by the number of times the monitor model is simulated. In particular, in the iterative cycle of the SSA it is calculated 30% more times than the HBA and the DOA.

For future works, different paths can be followed. The most banal would be to search for even more recent algorithms and implement the same failures to make

a comparison with the results obtained in this work. Or, one could focus on the algorithms used in this thesis and try to review their implementation perhaps using improved versions proposed in the literature. Even hybrid algorithms, although more complex, offer themselves as excellent solvers of optimization problems.

Once the algorithm has been chosen, before carrying out all the planned simulations, it could be interesting to carry out a trade-off analysis to understand the number of iterations that leads to convergence of the result, without imposing too much low error tolerance or too high maximum number of iterations. Furthermore, to evaluate the approximation capabilities of the algorithm, one could think of working directly with the RMSE which is the measure of how close the monitor model is to the reference model.

Whichever path you decide to pursue, it is useful to remember that the objective of the research is to find the most precise prognostic strategy possible and which can be carried out in reasonable operating times.

Bibliography

[1] I. Moir and A. G. Seabridge. *Aircraft systems: mechanical, electrical, and avionics subsystems integration*. 3rd ed. Aerospace series. OCLC: ocn190786124¦ Chichester, West Sussex, England ; Hoboken, NJ: Wiley, 2008. 504 pp. ISBN: 978-0-470-05996-8.

[2] P. Maggiore. *"Comandi di volo"*. Bachelor's degree in Aerospace engineering. Politecnico di Torino, 2019.

[3] Leonardo Baldo, Ivana Querques, Matteo Davide Lorenzo Dalla Vedova, and Paolo Maggiore. «A Model-Based Prognostic Framework for Electromechanical Actuators Based on Metaheuristic Algorithms». In: *Aerospace* 10.3 (Mar. 16, 2023), p. 293. ISSN: 2226-4310. (Visited on 04/16/2023).

[4] Gaetano Quattrocchi, Pier C. Berri, Matteo D. L. Dalla Vedova, and Paolo Maggiore. «An Improved Fault Identification Method for Electromechanical Actuators». In: *Aerospace* 9.7 (June 25, 2022), p. 341. ISSN: 2226-4310. DOI: 10.3390/aerospace9070341. URL: https://www.mdpi.com/2226-4310/9/7/341 (visited on 04/16/2023).

[5] Giampaolo Buticchi, Pat Wheeler, and Dushan Boroyevich. «The More-Electric Aircraft and Beyond». In: *Proceedings of the IEEE* 111.4 (Apr. 2023), pp. 356–370. ISSN: 0018-9219, 1558-2256. DOI: 10.1109/JPROC.2022.3152995. URL: https://ieeexplore.ieee.org/document/9732176/ (visited on 04/19/2023).

[6] Antonio Carlo Bertolino, Andrea De Martin, Giovanni Jacazio, and Massimo Sorli. «Design and Preliminary Performance Assessment of a PHM System for Electromechanical Flight Control Actuators». In: *Aerospace* 10.4 (Mar. 28, 2023), p. 335. ISSN: 2226-4310. DOI: 10.3390/aerospace10040335. URL: https://www.mdpi.com/2226-4310/10/4/335 (visited on 04/16/2023).

[7] L Baldo. «Prognostics of aerospace electromechanical actuators: comparison between model-based metaheuristic methods». In: ().

[8] D.L. Matteo, Dalla Vedova, Pier Carlo Berri, and Omayma Aksadi. «A novel model-based metaheuristic method for prognostics of aerospace electromechanical actuators equipped with PMSM». In: *IOP Conference Series: Materials Science and Engineering* 1226.1 (Feb. 1, 2022), p. 012107. ISSN: 1757-8981, 1757-899X. DOI: 10.1088/1757-899X/1226/1/012107. URL: https://iopscience.iop.org/article/10.1088/1757-899X/1226/1/012107 (visited on 04/16/2023).

[9] Jeremy Roussel, Marc Budinger, and Laurent Ruet. «Preliminary Sizing of the Electrical Motor and Housing of Electromechanical Actuators Applied on the Primary Flight Control System of Unmanned Helicopters». In: *Aerospace* 9.9 (Aug. 25, 2022), p. 473. ISSN: 2226-4310. DOI: 10.3390/aerospace9090473. URL: https://www.mdpi.com/2226-4310/9/9/473 (visited on 04/16/2023).

[10] Enrico Zio. «Prognostics and Health Management (PHM): Where are we and where do we (need to) go in theory and practice». In: *Reliability Engineering & System Safety* 218 (Feb. 2022), p. 108119. ISSN: 09518320. DOI: 10.1016/j.ress.2021.108119. URL: https://linkinghub.elsevier.com/retrieve/pii/S0951832021006153 (visited on 04/27/2023).

[11] M. Mazzoleni, Y. Maccarana, F. Previdi, G. Pispola, M. Nardi, F. Perni, and S. Toro. «Development of a reliable electro-mechanical actuator for primary control surfaces in small aircrafts». In: *2017 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*. 2017 IEEE International Conference on Advanced Intelligent Mechatronics (AIM). Munich, Germany: IEEE, July 2017, pp. 1142–1147. ISBN: 978-1-5090-5998-0 978-1-5090-6000-9. DOI: 10.1109/AIM.2017.8014172. URL: http://ieeexplore.ieee.org/document/8014172/ (visited on 05/10/2023).

[12] Mojtaba Kordestani, Marcos E. Orchard, Khashayar Khorasani, and Mehrdad Saif. «An Overview of the State of the Art in Aircraft Prognostic and Health Management Strategies». In: *IEEE Transactions on Instrumentation and Measurement* 72 (2023), pp. 1–15. ISSN: 0018-9456, 1557-9662. DOI: 10.1109/TIM.2023.3236342. URL: https://ieeexplore.ieee.org/document/10015835/ (visited on 04/28/2023).

[13] Kavindu Ranasinghe, Roberto Sabatini, Alessandro Gardi, Suraj Bijjahalli, Rohan Kapoor, Thomas Fahey, and Kathiravan Thangavel. «Advances in Integrated System Health Management for mission-essential and safety-critical aerospace applications». In: *Progress in Aerospace Sciences* 128 (Jan. 2022), p. 100758. ISSN: 03760421. DOI: 10.1016/j.paerosci.2021.100758. URL: https://linkinghub.elsevier.com/retrieve/pii/S0376042121000622 (visited on 04/27/2023).

[14] Zhengyang Yin, Niaoqing Hu, Jiageng Chen, Yi Yang, and Guoji Shen. «A review of fault diagnosis, prognosis and health management for aircraft electromechanical actuators». In: *IET Electric Power Applications* 16.11 (Nov. 2022), pp. 1249–1272. ISSN: 1751-8660, 1751-8679. DOI: 10.1049/elp2.12225. URL: https://onlinelibrary.wiley.com/doi/10.1049/elp2.12225 (visited on 03/19/2023).

[15] Matteo D. L. Dalla Vedova, Pier Carlo Berri, and Stefano Re. «A Comparison of Bio-Inspired Meta-Heuristic Algorithms for Aircraft Actuator Prognostics». In: *Proceedings of the 29th European Safety and Reliability Conference (ESREL)*. Proceedings of the 29th European Safety and Reliability Conference (ESREL). Research Publishing Services, 2019, pp. 1064–1071. ISBN: 978-981-11-2724-3. DOI: 10.3850/978-981-11-2724-3_0476-cd. URL: http://rpsonline.com.sg/proceedings/9789811127243/html/0476.xml (visited on 03/17/2023).

[16] Guan Qiao, Geng Liu, Zhenghong Shi, Yawen Wang, Shangjun Ma, and Teik C Lim. «A review of electromechanical actuators for More/All Electric aircraft systems». In: *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 232.22 (Nov. 2018), pp. 4128–4151. ISSN: 0954-4062, 2041-2983. DOI: 10.1177/0954406217749869. URL: http://journals.sagepub.com/doi/10.1177/0954406217749869 (visited on 04/21/2023).

[17] Pascal Vrignat, Frédéric Kratz, and Manuel Avila. «Sustainable manufacturing, maintenance policies, prognostics and health management: A literature review». In: *Reliability Engineering & System Safety* 218 (Feb. 2022), p. 108140. ISSN: 09518320. DOI: 10.1016/j.ress.2021.108140. URL: https://linkinghub.elsevier.com/retrieve/pii/S095183202100630X (visited on 04/27/2023).

[18] P. Nunes, J. Santos, and E. Rocha. «Challenges in predictive maintenance – A review». In: *CIRP Journal of Manufacturing Science and Technology* 40 (Feb. 2023), pp. 53–67. ISSN: 17555817. DOI: 10.1016/j.cirpj.2022.11.004. URL: https://linkinghub.elsevier.com/retrieve/pii/S1755581722001742 (visited on 03/19/2023).

[19] M. Fioriti. *"Gestione dei rischi, costi e supporto logistico integrato dei sistemi aerospaziali"*. Master's degree in Aerospace engineering. Politecnico di Torino, 2020.

[20] P. Maggiore; M. Dalla Vedova. *"Modellazione, simulazione e sperimentazione dei sistemi aerospaziali"*. Master's degree in Aerospace engineering. Politecnico di Torino, 2022.

[21] Sunday Ochella, Mahmood Shafiee, and Fateme Dinmohammadi. «Artificial intelligence in prognostics and health management of engineering systems». In: *Engineering Applications of Artificial Intelligence* 108 (Feb. 2022), p. 104552. ISSN: 09521976. DOI: 10.1016/j.engappai.2021.104552. URL: https://linkinghub.elsevier.com/retrieve/pii/S0952197621003961 (visited on 04/27/2023).

[22] Edward Balaban, Prasun Bansal, Paul Stoelting, Abhinav Saxena, Kai F. Goebel, and Simon Curran. «A diagnostic approach for electro-mechanical actuators in aerospace systems». In: *2009 IEEE Aerospace conference*. 2009 IEEE Aerospace conference. Big Sky, MT, USA: IEEE, Mar. 2009, pp. 1–13. ISBN: 978-1-4244-2621-8. DOI: 10.1109/AERO.2009.4839661. URL: http://ieeexplore.ieee.org/document/4839661/ (visited on 05/10/2023).

[23] M. Battipede. *"Guida e controllo del velivolo"*. Master's degree in Aerospace engineering. Politecnico di Torino, 2022.

[24] Mathworks. *Matlab documentation*. 2023. URL: https://it.mathworks.com/help/matlab/ref/floor.html?searchHighlight=floor&s_tid=srchtitle_floor_1.

[25] «AN885, Brushless DC (BLDC) Motor Fundamentals». In: (2003).

[26] Ivana Querques. «Prognostics of on-board electromechanical actuators: Bio-Inspired Metaheuristic Algorithms». PhD thesis. Politecnico di Torino, 2021.

[27] Paolo Maggiore. «Study of innovative model-based prognostic algorithms applied to aerospace electromechanical actuators». In: ().

[28] Brian Armstrong-Helouvry. «Stick slip and control in low-speed motion». In: *IEEE Transactions on Automatic Control* 38.10 (1993), pp. 1483–1496.

[29] Ricardo Solís, Lizeth Torres, and Pablo Pérez. «Review of Methods for Diagnosing Faults in the Stators of BLDC Motors». In: *Processes* 11.1 (Dec. 28, 2022), p. 82. ISSN: 2227-9717. (Visited on 08/21/2023).

[30] D Belmonte. «New Prognostic Method Based on Spectral Analysis Techniques Dealing with Motor Static Eccentricity for Aerospace Electromechanical Actuators». In: 14 (2015).

[31] Daniel Molina, Javier Poyatos, Javier Del Ser, Salvador García, Amir Hussain, and Francisco Herrera. «Comprehensive Taxonomies of Nature- and Bio-inspired Optimization: Inspiration Versus Algorithmic Behavior, Critical Analysis Recommendations». In: *Cognitive Computation* 12.5 (Sept. 2020), pp. 897–939. ISSN: 1866-9956, 1866-9964. DOI: 10.1007/s12559-020-09730-8. URL: https://link.springer.com/10.1007/s12559-020-09730-8 (visited on 08/27/2023).

[32] G. Devika and Asha Gowda Karegowda. «Bio-inspired Optimization: Algorithm, Analysis and Scope of Application». In: *Artificial Intelligence*. Ed. by Marco Antonio Aceves-Fernández. Vol. 15. IntechOpen, Feb. 8, 2023. ISBN: 978-1-83768-086-3 978-1-83768-087-0. DOI: 10.5772/intechopen.106014. URL: https://www.intechopen.com/chapters/84300 (visited on 03/17/2023).

[33] Rajiv Yadav, Indu Sreedevi, and Daya Gupta. «Bio-Inspired Hybrid Optimization Algorithms for Energy Efficient Wireless Sensor Networks: A Comprehensive Review». In: *Electronics* 11.10 (May 12, 2022), p. 1545. ISSN: 2079-9292. DOI: 10.3390/electronics11101545. URL: https://www.mdpi.com/2079-9292/11/10/1545 (visited on 03/10/2023).

[34] Zahra Beheshti and Siti Mariyam Hj Shamsuddin. «A Review of Population-based Meta-Heuristic Algorithm». In: ().

[35] «Initialisation Approaches for Population-Based Metaheuristic Algorithms: A Comprehensive Review». In: *Applied Sciences* 12 (). ISSN: 2076-3417. DOI: 10.3390/app12020896. URL: https://www.mdpi.com/2076-3417/12/2/896.

[36] Thomas Bäck. «Organic Evolution and Problem Solving». In: *Evolutionary Algorithms in Theory and Practice*. Oxford University Press, Feb. 15, 1996. ISBN: 978-0-19-509971-3 978-0-19-756092-1. DOI: 10.1093/oso/9780195099713.003.0006. URL: https://academic.oup.com/book/40791/chapter/348728584 (visited on 08/27/2023).

[37] Jiankai Xue and Bo Shen. «A novel swarm intelligence optimization approach: sparrow search algorithm». In: *Systems Science & Control Engineering* 8.1 (Jan. 1, 2020), pp. 22–34. ISSN: 2164-2583. DOI: 10.1080/21642583.2019.1708830. URL: https://www.tandfonline.com/doi/full/10.1080/21642583.2019.1708830 (visited on 04/12/2023).

[38] Farhad Soleimanian Gharehchopogh, Mohammad Namazi, Laya Ebrahimi, and Benyamin Abdollahzadeh. «Advances in Sparrow Search Algorithm: A Comprehensive Survey». In: *Archives of Computational Methods in Engineering* 30.1 (Jan. 2023), pp. 427–455. ISSN: 1134-3060, 1886-1784. DOI: 10.1007/s11831-022-09804-w. URL: https://link.springer.com/10.1007/s11831-022-09804-w (visited on 04/12/2023).

[39] Narinder Singh and S. B. Singh. «Hybrid Algorithm of Particle Swarm Optimization and Grey Wolf Optimizer for Improving Convergence Performance». In: *Journal of Applied Mathematics* 2017 (2017), pp. 1–15. ISSN: 1110-757X, 1687-0042. DOI: 10.1155/2017/2030489. URL: https://www.hindawi.com/journals/jam/2017/2030489/ (visited on 04/11/2023).

[40] Mohammed A. Awadallah, Mohammed Azmi Al-Betar, Iyad Abu Doush, Sharif Naser Makhadmeh, and Ghazi Al-Naymat. «Recent Versions and Applications of Sparrow Search Algorithm». In: *Archives of Computational Methods in Engineering* (Feb. 7, 2023). ISSN: 1134-3060, 1886-1784. DOI: 10.1007/s11831-023-09887-z. URL: https://link.springer.com/10.1007/s11831-023-09887-z (visited on 04/12/2023).

[41] Fatma A. Hashim, Essam H. Houssein, Kashif Hussain, Mai S. Mabrouk, and Walid Al-Atabany. «Honey Badger Algorithm: New metaheuristic algorithm for solving optimization problems». In: *Mathematics and Computers in Simulation* 192 (Feb. 2022), pp. 84–110. ISSN: 03784754. DOI: 10.1016/j.matcom.2021.08.013. URL: https://linkinghub.elsevier.com/retrieve/pii/S0378475421002901 (visited on 04/14/2023).

[42] Jianan Lin, Rongjia Zheng, Yirong Zhang, Jinkai Feng, Wei Li, and Kaiqing Luo. «CFHBA-PID Algorithm: Dual-Loop PID Balancing Robot Attitude Control Algorithm Based on Complementary Factor and Honey Badger Algorithm». In: *Sensors* 22.12 (June 14, 2022), p. 4492. ISSN: 1424-8220. DOI: 10.3390/s22124492. URL: https://www.mdpi.com/1424-8220/22/12/4492 (visited on 04/14/2023).

[43] Muhammad Haris Khan, Abasin Ulasyar, Abraiz Khattak, Haris Sheh Zad, Mohammad Alsharef, Ahmad Aziz Alahmadi, and Nasim Ullah. «Optimal Sizing and Allocation of Distributed Generation in the Radial Power Distribution System Using Honey Badger Algorithm». In: *Energies* 15.16 (Aug. 14, 2022), p. 5891. ISSN: 1996-1073. DOI: 10.3390/en15165891. URL: https://www.mdpi.com/1996-1073/15/16/5891 (visited on 04/14/2023).

[44] Abdelfattah Elhammoudy, Mustapha Elyaqouti, El Hanafi Arjdal, Dris Ben Hmamou, Souad Lidaighbi, Driss Saadaoui, Imade Choulli, and Ismail Abazine. «Dandelion Optimizer algorithm-based method for accurate photovoltaic model parameter identification». In: *Energy Conversion and Management: X* 19 (July 2023), p. 100405. ISSN: 25901745. DOI: 10.1016/j.ecmx.2023.100405. URL: https://linkinghub.elsevier.com/retrieve/pii/S2590174523000612 (visited on 09/19/2023).

[45] *Dandelion Optimizer - File Exchange - MATLAB Central*. URL: https://it.mathworks.com/matlabcentral/fileexchange/114680-dandelion-optimizer (visited on 09/19/2023).

[46] Shijie Zhao, Tianran Zhang, Shilin Ma, and Miao Chen. «Dandelion Optimizer: A nature-inspired metaheuristic algorithm for engineering applications». In: *Engineering Applications of Artificial Intelligence* 114 (), p. 105075. ISSN: 09521976. DOI: 10.1016/j.engappai.2022.105075. URL: https://linkinghub.elsevier.com/retrieve/pii/S0952197622002305.

[47] Timothy O. Hodson. «Root-mean-square error (RMSE) or mean absolute error (MAE): when to use them or not». In: *Geoscientific Model Development* 15.14 (July 19, 2022), pp. 5481–5487. ISSN: 1991-9603. DOI: 10.5194/gmd-15-5481-2022. URL: https://gmd.copernicus.org/articles/15/5481/2022/ (visited on 08/29/2023).

[48] *MATLAB Performance - MATLAB & Simulink*. URL: https://it.mathworks.com/products/matlab/performance.html (visited on 10/02/2023).