Politecnico di Torino
1859

ScuDo
Scuola di Dottorato ~ Doctoral School
WHAT YOU ARE, TAKES YOU FAR

Doctoral Dissertation
Doctoral Program in Aerospace Engineering (35th cycle)

Development of optical sensors and diagnostics algorithms for aerospace systems

By

Gaetano Quattrocchi

******

Supervisor(s):
Prof. P. Maggiore, Supervisor
Dr. M.D.L. Dalla Vedova, Co-Supervisor
Prof. D. Janner, Co-Supervisor

Doctoral Examination Committee:
Prof. F. Tornabene, Università degli Studi del Salento
Prof. E. Minisci, University of Strathclyde
Prof. N. Garzaniti, Cranfield University
Prof. D. Milanese, Università degli Studi di Parma

Politecnico di Torino
2023

Declaration

I hereby declare that, the contents and organization of this dissertation constitute my own original work and does not compromise in any way the rights of third parties, including those relating to the security of personal data.

Gaetano Quattrocchi
2023

* This dissertation is presented in partial fulfillment of the requirements for **Ph.D. degree** in the Graduate School of Politecnico di Torino (ScuDo).

To Giorgia, once more.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A steady push towards more feature-rich, integrated and thus complex systems has been observed, in recent years, in the aerospace sector. This trend can be explained given the many advantages presented by a *smart system*, i. e. a system capable of autonomously performing corrective or even predictive actions when particular conditions occur. In a broader sense, aerospace systems can be generally thought as smart given the very high safety and reliability standards imposed, which implies redundancies and active monitoring of the system status; mitigating strategies (e.g. passive or cold redundancy) are implemented should a critical event occur. However, traditional, cold redundancy based methods have drawbacks, namely reliability reduction, sub-optimal system utilization, weight and cost increase [1].

The logical evolution is the switch from a corrective to a predictive approach, i.e. performing a mitigating action before a critical event occurs. In this context, the Prognostics and Health Management (PHM) [2] framework is very useful. In brief, the objective of PHM is to give an accurate estimation of the 'health status' of a system defined in term of prognostics indicators. Said indicators are variable and application-dependent, but the commonality is the embedding of status information and sensitivity to one or more faults that can occur in the system. The concept of Remaining Useful Life (RUL) [3] is important, as it is the main parameter that correlates the actual system health status and maintenance events.

In fact, using a PHM approach, Condition Based Maintenance (CBM) [4] can be achieved. While traditional, preventive maintenance is generally a result of safe-life philosophy, i.e. components are replaced after a predetermined amount of time, independently of the health status, a CBM approach schedules maintenance events when necessary by monitoring the condition of components or systems.

To perform health status monitoring, two approaches are generally used. The first approach is model-based, where either a physical model or a digital twin is used [5]. The other approach is data-driven, where machine learning methods are applied to data generated by the system to infer relations [6].

Another technology steadily gaining adoption is optical fiber sensors, such as Fiber Bragg Gratings (FBGs), which offers several advantages over traditional, electronic based sensors. FBGs can primarily sense strain by alteration of the reflected Bragg wavelength; this measure can be correlated with other physical quantities when using appropriate fiber coating. Many different FBG-based sensors exist and are used to measure humidity [7, 8], temperature [9–12], pressure [13–15], gas and pollutant concentrations [16–18] and more.

This work aims at integrating FBGs in a diagnostics framework which uses an electromechanic actuator (EMA) as a real test-case. The strain measures measured using FBGs will be used in a machine-learning algorithm to estimate the health status of the system. Additionally, a parallel evaluation can be made for other faults such as electrical partial shorts and static eccentricity. Additionally, several FBG-based sensor designs will be proposed; these sensors could be used as sources of information in the prognostic process. Finally, several reduced order model with varying levels of fidelity will be presented given their usefulness in reducing the simulations computational time for use cases such as real-time monitoring.

Chapter 2

Optical fibers and sensors

In this section, the operating principles of fiber optics and Bragg gratings will be described. Furthermore, some new sensors designs will be reported and characterized.

2.1 Optical fibers overview

Optical fibers are flexible and thin strands of glass (or plastic) capable of transmitting light, not necessarily in the visible spectrum. They are a key technology in telecommunications, composing the backbone of the Internet infrastructure. Due to extremely low attenuation, a signal in a fiber optic can be transmitted for kilometers or tens of kilometers before needing amplification. Fibers are also immune to electro-magnetic interference as opposed to normal copper cables.

Furthermore, bandwidth is several order of magnitude greater than traditional copper cables (hundreds of Gbps vs hundreds of Mbps). Recent lab testing managed to push the maximum data-rate to petabit per second [19]. Optical fibers are also much lighter and flexible than copper cables.

2.2 Principles of operation

Optical fibers are circular waveguides that bounds the propagation of light inside it. Generally, the wavelength of light used is chosen as to minimize the absorption during propagation, thus reducing attenuation as much as possible. For example, in

Fig. 2.1 [20], the attenuation spectrum for normal fused silica glass optical fibers is shown.

The three optical or transmission windows (shown as colored bands) are valleys in the attenuation plot as function of wavelength. The lowest value is about 0.2 dB/km and is obtained at around 1550 nm, which corresponds to short wave infrared.

The band between 1500 and 1600 nm is usually designed third telecom window, and presents the lowest absorption region in the whole spectrum for silica glass (blue band in Fig. 2.1). The two other transmission windows, at shorter wavelengths, are respectively the 800-1000 nm band (red) which is historically the first band used, thus called the first telecom window. Finally, in green, the second telecom window is shown, at around 1300nm of wavelength.

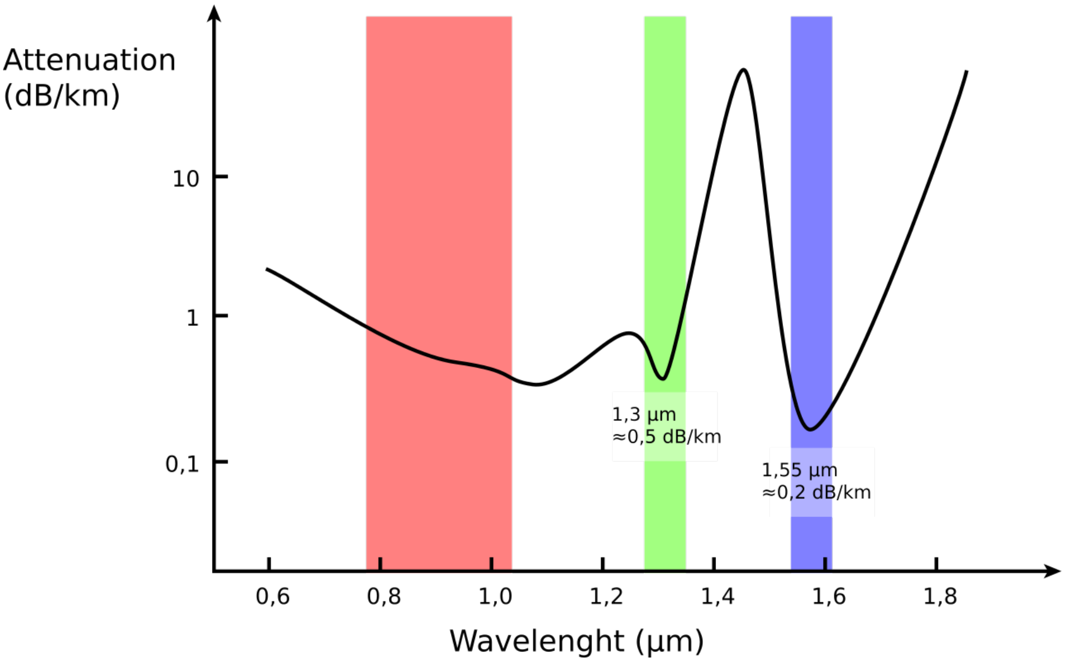


Fig. 2.1 Optical windows for silica glass

The physical phenomena that allows optical fibers to be waveguide is total internal reflection, which directly postulates from Snell's Law:

$$\sin\theta_1 n_1 = \sin\theta_2 n_2 \tag{2.1}$$

where θ_1 is the angle of incidence, θ_2 is the angle of refraction and $n_{1,2}$ are the refraction indexes of the two material. The refraction index is defined as the ratio of the speed of light in vacuum (c) over the speed of light in the medium (v):

$$n = \frac{c}{v} \tag{2.2}$$

Snell's law applies every time there is an interface along the propagation of the wave. As visible in Fig.2.2 [21], the angle of incidence is 60^o; the reflected beam forms the same angle. The refracted ray, however, has a different angle with respect to the normal, which can be determined using Eq. 2.1.

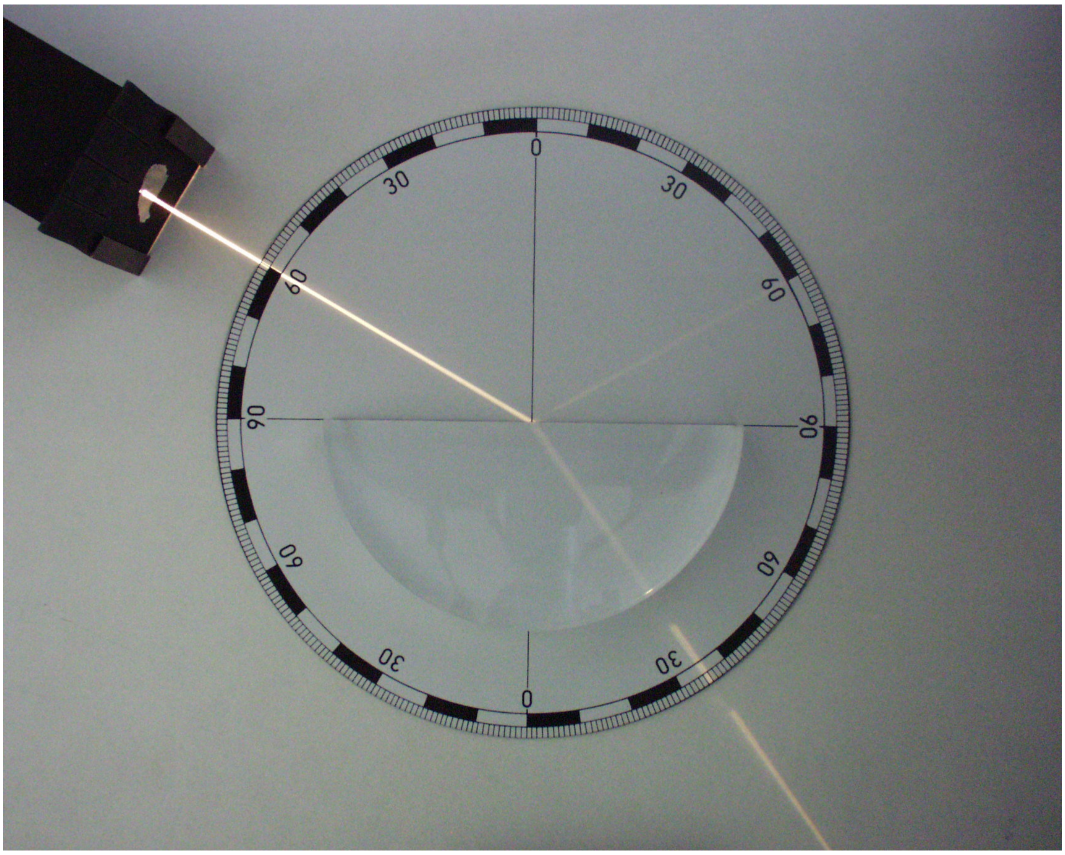


Fig. 2.2 Snell's Law example

A condition that can occur when $n_1 > n_2$, i.e. when there is a passage from a denser to a less dense medium, is called total internal reflection. From Eq. 2.1, one can find that the equation has no solutions when θ_1 is greater than a value called *critical angle*, which can be obtained as:

$$\theta_{crit} = \arcsin\left(\frac{n_2}{n_1}\right) \tag{2.3}$$

If the angle of incidence is greater than θ_{crit}, no refraction will occur and total internal reflection will be observed. This particular phenomenon is key for the operation of an optical fiber, since it bounds the light inside it, without loss of intensity along the fibers path.

2.2.1 Single-mode and multi-mode fibers

One of the possible classifications of optical fibers can be made on the modes that a fiber can propagate. If only one mode can be propagated, then the fiber is defined as single-mode fiber, otherwise it is called multi-mode.

Differences between the two types of fibers are both optical and physical, as shown in Fig. 2.3 (image from [22]).

Single-mode fibers have small cores, in the order of 8-10 μm, with external diameters of usually 125 μm. The main advantage of a single mode fiber is the lack of intermodal dispersion, arising from the difference in optical path length which is problematic in multi-mode fibers.

Multi-mode fibers have larger cores, in the order of tens of microns, and can propagate several or many optical modes. Intermodal dispersion is always present in these fibers, accompanied to pulse broadening phenomena [23]. One mitigating strategy is the adoption of graded index over step index fibers, which reduces dispersion limiting the penetration of higher order modes electric fields in the cladding.

Another difference is the ease of coupling of the fiber with the emitting source, being it a LED or a laser. A parameter often used to measure the accepting cone is the numerical aperture (NA), which is defined as:

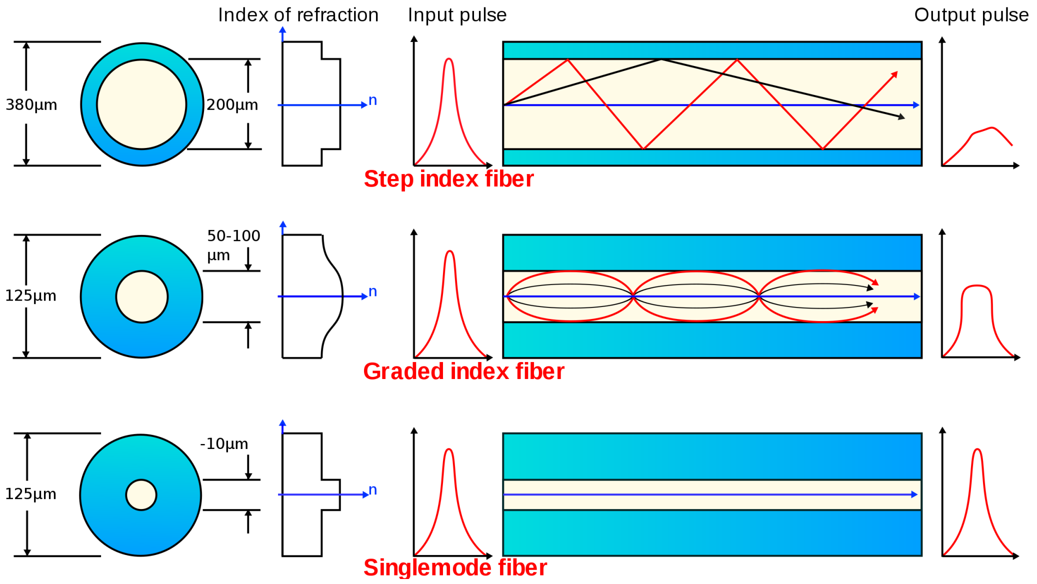


Fig. 2.3 Schematics of single and multi mode fibers

$$NA = \sqrt{n_{core}^2 - n_{clad}^2} \tag{2.4}$$

The numerical aperture is related to the acceptance angle in air with the following:

$$\theta_a = \arcsin(NA) \tag{2.5}$$

However, this approximation is only valid for multi-mode fibers and is not applicable to single mode fibers. In any case, typical values are of about 0.1 but can vary from 0.05 to 0.4 for single mode fibers, while typical values for multi-mode are around 0.3.

2.2.2 Materials for optical fibers

The most used material for optical fibers is *silica*, i.e. silicon dioxide (SiO_2), as it presents good optical transmission properties along a wide band of the EM spectrum. As previously exemplified in Fig. 2.1, absorption and scattering losses can be as low as about 0.2 dB/km at around 1550 nm.

However, such low attenuations requires extremely pure silica, as even minuscule impurities can lower the transmission properties by order of magnitudes. For example, at around 1.4 μ, the absorption peak of the -OH group is always present, but it can be lowered down by reducing the free hydroxyl groups present in the glass, being one of the overtones of the -OH group as shown in [24].

In any case, the absorption around 1400 nm is about two orders of magnitude higher than at around 1550 nm.

Another advantage of silica is the good mechanical capabilities it exhibits, having an extremely high tension failure stress (more than 5000 MPa [23]) and moderate bending resistance. Furthermore, silica is chemically inert and is not hygroscopic.

Silica can be doped with different oxides to change the refractive index; aluminium or germanium oxide (Al_2O_3, GeO_2) are commonly used to raise the refractive index, while boron oxide (B_2O_3) or fluorine F_2 are both used to lower it, as shown in 2.4 (image from [25]). However, low solubility for rare-earth elements has been observed so silica isn't very apt for active fibers elements such as amplifier or lasers.

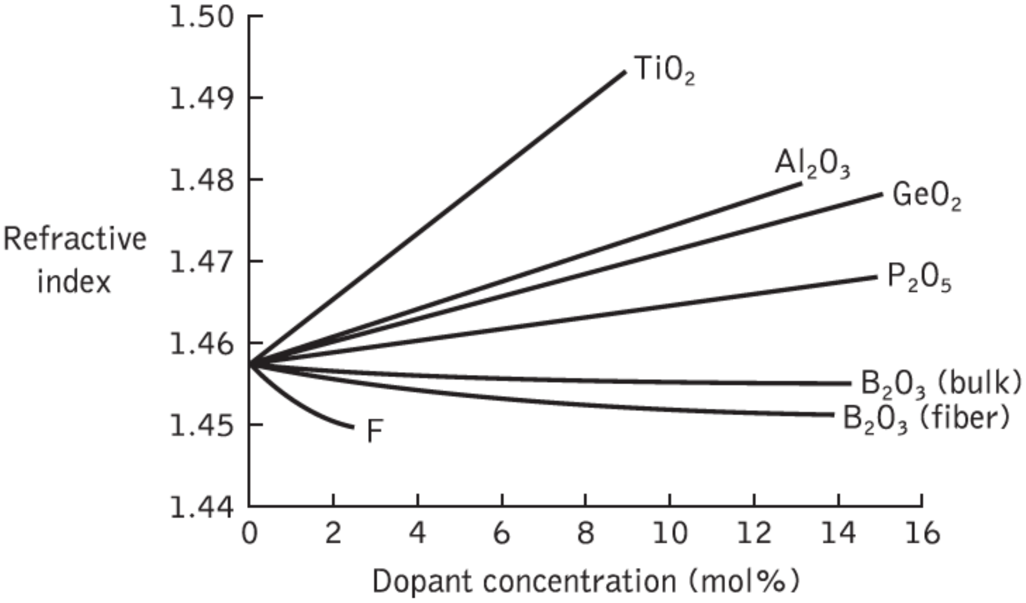


Fig. 2.4 Several dopants used to alter refraction index

Other, much less used classes of glasses are fluoride and phosphate-based. Fluoride glass have several disadvantages discouraging mass adoption, such as hard to avoid partial crystallization, fragility and poor environmental stability; the main advantage is the lack of hydroxyl-associated absorption band at 2780-3125 nm as fluorides are non-oxide based.

Phospate glass are composed my (meta)phospates of metals; analogously to silicate, a tetrahedral structure is observed in phospates glass, where the elemental cell is composed by phosporous pentoxide P_2O_5. Four different polymorphic structures exist, with the most popular being P_4O_{10}.

Phospate glasses have good mechanical properties (e.g. tensile strength around 2000 MPa [26]) and show a high solubility for rare-earth elements ions, making them useful in active optical components.

Other types of glasses, such as chalcogenide glass exist, and are made with chalcogens elements (group 16 of the periodic table, such as sulfur (S), tellurium (Te) and selenium (Se)) reacting with metals such as silver (Ag). In general, chalcogenide glasses are very versatile, since most of the properties can be tailored to the desired use case, such as crystalline or amorphous structure, band gap width and ions/electrons conductivity.

Finally, plastics are also used in optical fibers, in the so called POFs - Platic Optic Fibers. The most common material used for core is either PMMA (poly(methyl methacrylate)) or polystyrene, while silicone is generally used for cladding. POFs offer several advantages over glass-based fibers, such as much lower cost both for fibers and supporting equipment, good mechanical flexibility and high numerical aperture. In general, the core is much bigger than in glass fibers, in the order of 1mm compared to tens of microns. However, POFs have much higher absorption compared to glass fibers and are thus limited to short-range communications, such as in home Gigabit [27]. POFs are also used in the making of fixtures such as lamps.

2.2.3 Manufacturing

To obtain the level of precision needed to create an optical fiber, ad-hoc manufacturing methods have been developed.

The first step of glass optical fibers manufacturing is the production of a preform, which is a large diameter element with a very precise index of refraction profile, which is then stretched to form the fiber strands.

Several techniques are used, which are all different types of chemical vapor deposition (CVD): outside vapor deposition, inside vapor deposition and axial vapor deposition.

In inside vapor deposition, shown in Fig. 2.5 (from Ref. [28]), gases such as silicon or germanium tetrachloride ($SiCl_4$, $GeCl_4$) are injected inside a glass tube, mixed with oxygen and burned in gas phase; the products of the reactions, such as silica, will then deposit on the inside wall as soot; an external oxyhydrogen torch is used to maintain the inside temperature to around 1500/1600 °C, allowing the reaction to proceed. The flame is also moved along the preform direction to even the reaction conditions and to allow the deposited soot to melt and form a new layer. The technique is very versatile allowing precise control on thickness and composition (and thus index of refraction) of each layer, by varying temperature and/or chemical composition of the inbound gas mixture.

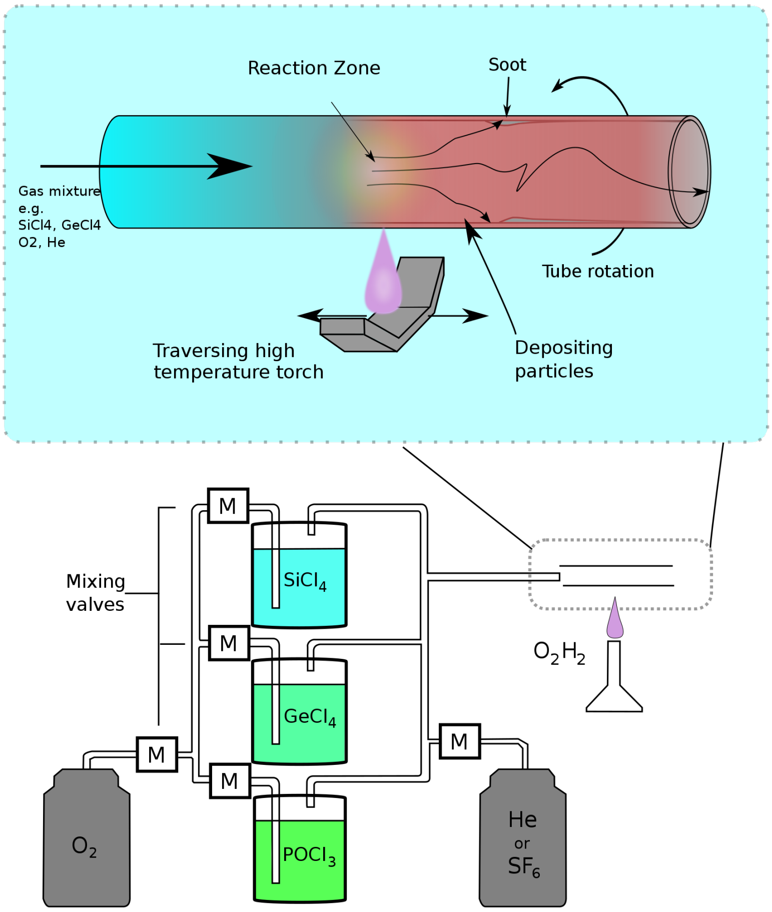


Fig. 2.5 Process of inside vapor deposition

On the other hand, in outside or axial deposition, glass is formed on the external surface of a seed rod, by combusting silicon or germanium tetrachloride in an oxyhydrogen flame; the difference between the two techniques is the type of the seed rod used, which is either removed or used as starter element and axially built along, obtaining a porous preform. It is consolidated in a solid preform by heating at around 1500 °C.

After the preform has been manufactured, it needs to be drawn. Using an instrument calling drawing tower, which stretches the preform to the desired thickness, after heating the preform. At the same time, the coating is applied in one or more passes depending on the specific process. Finally, polymer coatings (generally PMMA or polyacrilate) is cured using UV lamps. The coating layers are needed to increase mechanical strength of the fiber and to improve the resistance to environmental conditions.

2.3 Optical sensors

Optical fibers can be used to measure various quantities such as strain, temperature and pressure. This is done by modifying the fiber so that the quantity being measured changes the light's intensity, phase, polarization, wavelength or transit time. Intensity-based sensors are the simplest as they only require a source and detector. Fiber-optic sensors can also provide distributed sensing over large distances if needed.

Temperature can be measured using a fiber with evanescent loss that changes with temperature or by analyzing one of several possible scattering modes (Rayleigh, Raman, Brillouin) of light in the fiber. Specially-doped fiber that alters the polarization of light based on voltage or electric field allows to directly measure voltages. Interferometric angle measurement sensors have been developed using the Sagnac effect, such as gyroscopes [29].

It is also possible to measure more than one quantity at the same time, for example strain and temperature, by using more than a single effect, for example in [30], using both FBG wavelength shift and Er^{3+} fluorescence.

Other applications include strain and temperature sensing in hostile environments such as in proximity of Magnetic Resonance machines [31], due to the extreme electromagnetic fields, or in oil wells [32], given the range of pressure and temperatures that does not allow the use of conventional electric sensors.

Sensing can be concentrated in specific points along the fiber, using sensors such as Fiber Bragg Gratings (FBGs), or distributed, leveraging one of the previously mentioned scattering effects; it has been shown that using Brillouin scattering it is possible to monitor a 250-km long stretch of fiber with a resolution of 5 m [33].

2.3.1 Fiber Bragg gratings

Fiber Bragg Gratings (FBGs) are some of the most commonly used optical sensor in different fields, such as oil and gas, medical and aerospace.

In essence, FBGs are periodical variations in the fiber's core refractive index, creating a very narrow band dielectric mirror. The principle of operation is based on Fresnel reflection. An FBG has periodic alteration of the index of refraction of the core of the fiber, which create reflected waves at each change. The series of reflected waves is in general out of phase, except for a specific wavelength, defined as Bragg wavelength which correlates to the physical spacing of the variations, generally formulated as:

$$\lambda_B = 2n_e\Lambda \tag{2.6}$$

where n_e is the effective index of refraction of the core and Λ is the physical spacing of the variations of refraction index. n_e is function of the wavelength but also of the propagation mode in the fiber (for multimode fiber); it is thus only a function of the wavelength when using a single-mode fiber.

As shown in Fig.2.6 (from [34]), a wide band of wavelength is sent through the fiber where the FBG is present; the FBG works as a highly selective wavelength mirror, allowing most of the wavelengths to passa through while a narrow peak centered around the Bragg wavelength is reflected back and can be read.

The usefulness of FBGs is in their correlation between bands spacing and Bragg wavelength; this allows direct measurement of mechanical strains and temperature at the same time. In fact, one of the most used equations to correlates Bragg wwavelength variation to physical variation is the following:

$$\frac{\Delta\lambda_B}{\lambda_B} = (1 - p_e)\varepsilon + (\alpha_\Lambda + \alpha_n)\Delta T \tag{2.7}$$

where $\Delta\lambda_B$ is the variation of Bragg wavelength, p_e is the strain optic coefficient, ε is the strain applied to the FBG, α_Λ is the coefficient of thermal expansion of the fiber and α_n is the thermo-optic coefficient.

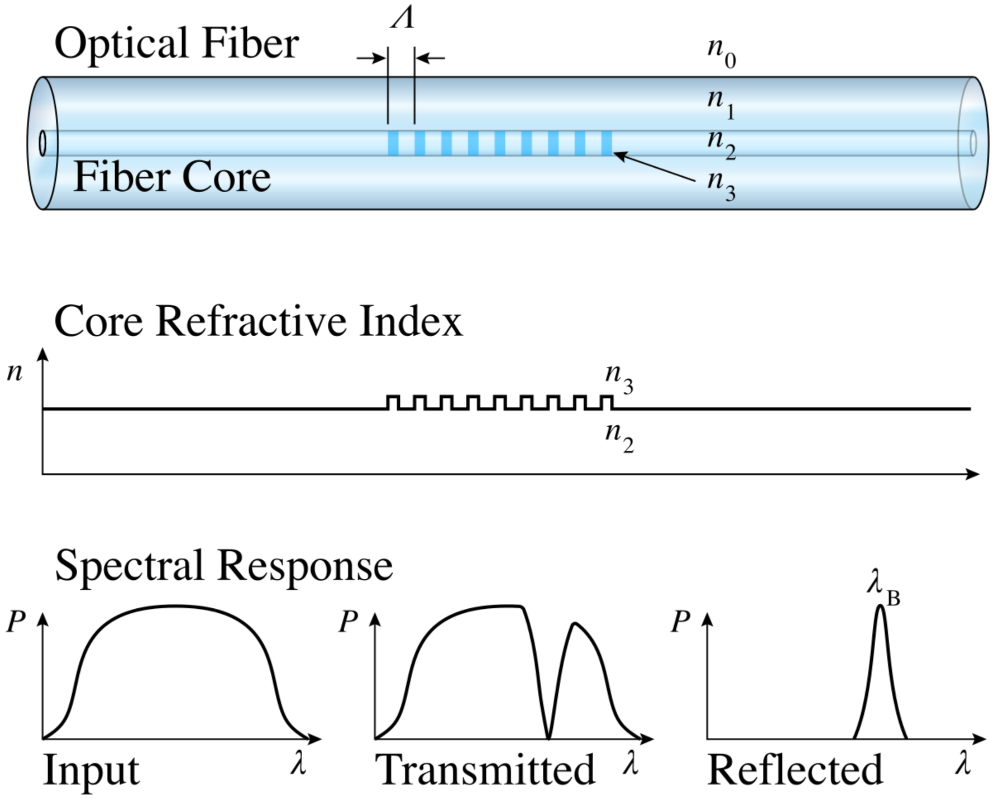


Fig. 2.6 FBG principle of operation

Typical values for silica fibers are $\alpha_\Lambda \simeq 0.55 \times 10^{-6}\ \ K^{-1}$ and $\alpha_n \simeq 8.6 \times 10^{-6}\ \ K^{-1}$, i.e. the thermo-optic coefficient has a much bigger effect on the thermal wavelength shift (in total, on the order of 14 pm/°C) and $p_e \simeq 0.83$ [35], i.e. a 1 $\mu\varepsilon$ ($1 \times 10^{-6}\Delta l/l$) strain induces a 1.2 pm wavelength shift. However, the FBG is sensitive to both strain and temperature at the same time, so it is useful when only on of the two measures changes over time; on the other hand, in most mechanical applications, it is not often that a measuring environment offers a constant temperature or a no vibration zone. In this, more general case, a decoupling of the two measures has to take place, either using hardware, software or multiple sensors using different coatings and thus having different temperature-wavelength or strain-wavelength shift curves.

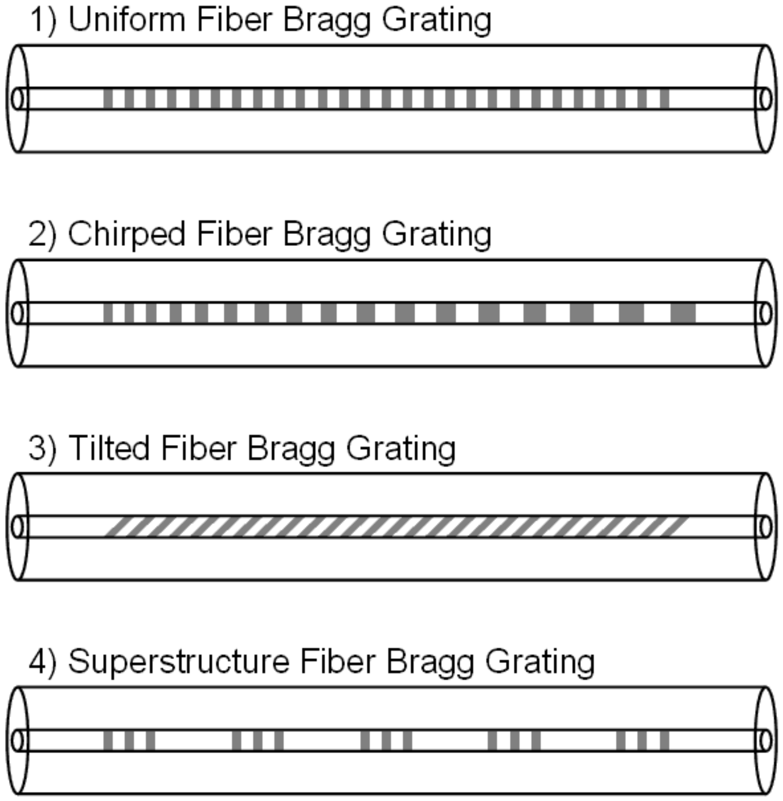

1) Uniform Fiber Bragg Grating
2) Chirped Fiber Bragg Grating
3) Tilted Fiber Bragg Grating
4) Superstructure Fiber Bragg Grating

Fig. 2.7 Several types of Bragg gratings structures

Several types of Bragg gratings exist, even tough the most common types found in sensing application is the so-called 'standard' or type-1 grating, originally developed for the telecommunications industry as dispersion compensators or WDM (wavelength division multiplexing) filters [36].

Other types of Bragg gratings are classified still as type-1(x), which x can be n (negative), H (hydrogen), Hp, Hs (hypersensitised) or d (densification) [36]. All the type-1 gratings are manufactured in either Si-based or Ge-doped fibers, using UV-laser interference in the fiber core. In any case, type-1 gratings are all written with energies under the glass damage threshold. Furthermore, type-1 gratings are generally made either on silica fibers or germania-doped fibers, depending on the variant. Additionally, substrate and nature of the gratings determine measure stability and the maximum operating temperature.

On the other hand, type-2 gratings are manufactured with extremely high energy pulses, obtained using for example femtosecond-lasers, as to reach and overcome the damage threshold in glass, leading to fracturing, void formation and possibly filamentation [36].

The other important parameter that can be tailored to the user needs is the grating structure. The simplest is the uniform index change, as shown in Fig. 2.7(1) (from [37]). Other common structures are chirped (2), tilted or with superstructures (4).

However, if one uses a simple discrete variation of index of refraction n, the reflected spectrum will not be a single peak, but it will contain secondary lobes, due to the step change of index. This effect can be mitigated using apodization, i.e. a modulation of the amplitude of the refraction index variation using a gaussian function, as shown in Fig. 2.8 (from [38]) . This allows reduction of the amplitude of the side-lobes of the reflected waveband, thus improving the returning signal quality.

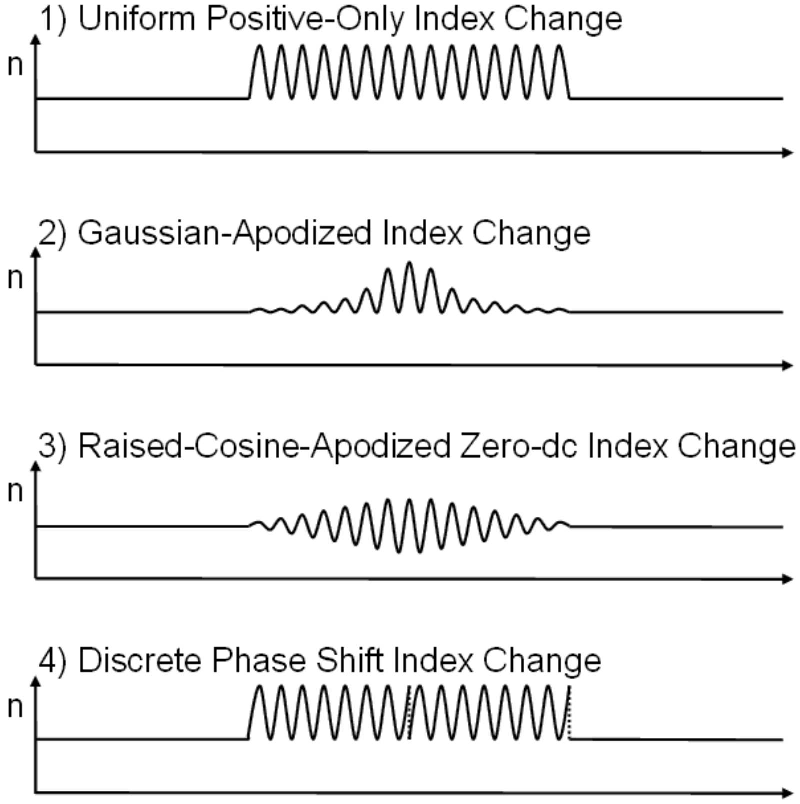

1) Uniform Positive-Only Index Change
n
2) Gaussian-Apodized Index Change
n
3) Raised-Cosine-Apodized Zero-dc Index Change
n
4) Discrete Phase Shift Index Change
n

Fig. 2.8 Possible gratings modulations

2.4 Developed sensors

During the PhD, several different sensors have been prototyped and tested, ranging from vibration sensors to temperature and strain sensors. As previously explained, a single FBG senses both temperature and strain at the same time, thus some methodology to decouple the measurements is necessary if one's interested in measuring only one of the quantities, since in most mechanical applications is hard to have a constant temperature environment or a vibration-free one.

However, for the first case presented, i.e. a vibration sensor, the temperature variation is not considered too important since amplitude measurements are relative and mainly the vibration frequency is of interest.

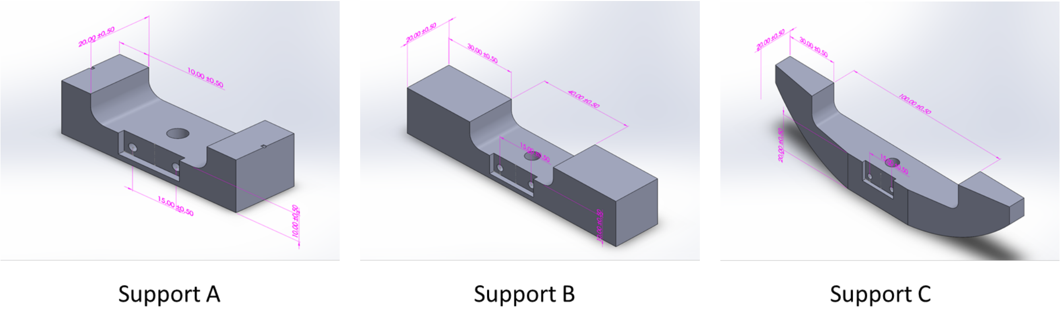
2.4.1 Vibration sensor

In literature, several solutions are commonly adopted such as bare fibers vibrating [39], cantilevers [40], orthogonal flexure hinges [41] or multi-axis flexure hinges [42]. In general, these works aims at having a resonant frequency out of the range of measuring, as to have the maximum linearity during measurements.

However, in our design, the goal is the opposite, i.e. manufacture a sensor that resonates at a frequency of interest; this enables to detect minute amplitude variations, which is useful in mechanical applications as many rotating machines tend to operate at a fixed frequency, thus often showing vibration peaks at frequencies corresponding to the rotational speed or at multiples of that when some faults start to appear. Part of this work has been presented in [43].

Initial development consisted in the manufacturing of three different 3D-printed supports made of PLA, as shown in Fig. 2.9, using fused deposition (FDM). Four different fibers were used in testing, since fiber #2 failed in early pre-testing (glued to support B), and thus another fiber (#4) was glued to support B, as reported in Table 2.1, where L is the fiber free length, and ω_n is the support combination natural frequency calculated by FEM (Finite Element Method).

Furthermore, for fibers #1 and #4, a reduction in pre-tensioning has been observed before testing, and it is probably caused by either the cyanoacrilate glue yielding or a permanent (microscopic) deformation of the support caused by the fiber stress, or a combination of both. In any case, since this facts have been observed before proper testing, the lower, measured value of pretensioning has been used during the testing campaign. Pre-tensioning can be measured by reading the Bragg wavelength of the sensor and comparing it to the nominal Bragg wavelength measured in no-stress conditions before gluing, thus obtaining the microstrains ($\mu\varepsilon$) estimation.



(a) First sample

(b) Second sample

(c) Third sample

Fig. 2.9 Series of samples tested

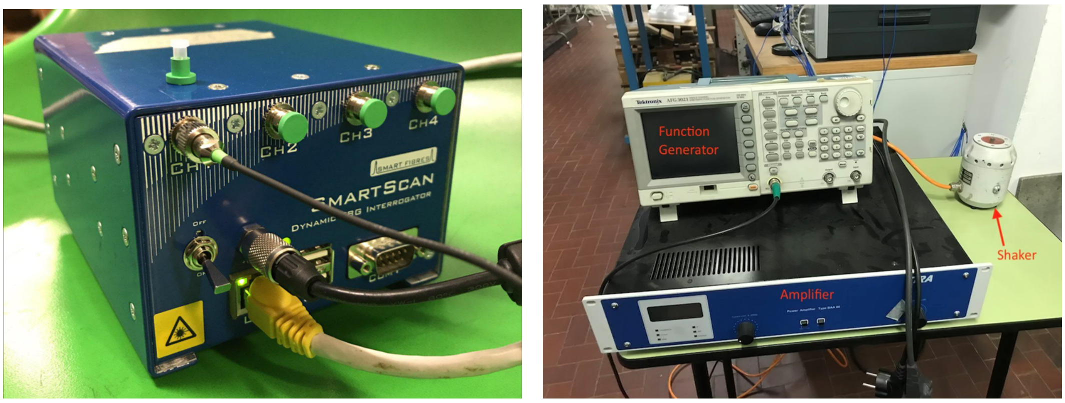
Combination	L [mm]	Pre-tension [$\mu\varepsilon$], appr..	ω_n^* [Hz], FEM
1 - A	40	1600 → 1000	1820
2 - B	40	500	794
3 - C	100	500	978
4 - B	40	4200 → 2200	794

Table 2.1 Fibers and supports data

For the whole testing campaign, the experimental apparatus shown in Fig. 2.9 has been used, consisting of a function generator, the output of which is amplified by an amplifier then connected to a mechanical shaker (Fig. 2.9a), and on the other side a SMARTSCAN FBG interrogator (Fig. 2.9b) which reads the output of the sensor. In Fig.2.11 a picture of sensor 3 is shown during a testing session, with the feedback accelerometer visible under the sensor, above the shaker (blue cable).

Testing results

Initially, some tests have been performed to validate the testing apparatus. In particular, Fig. 2.12 shows the FFT of a test performed at 350 Hz sinusoidal excitation; the data source is the interrogator and thus the sensor. The exciting frequency is clearly visible in the frequency spectrum, but numerous harmonics are also present, due to the non-linearity of the material. The second harmonics could be



(a) Function generator, amplifier and shaker (b) FBG interrogator

Fig. 2.10 Experimental setup overview

explained by the fiber behavior similar to a guitar string, vibrating (and deforming) at exactly twice the excitation frequency.

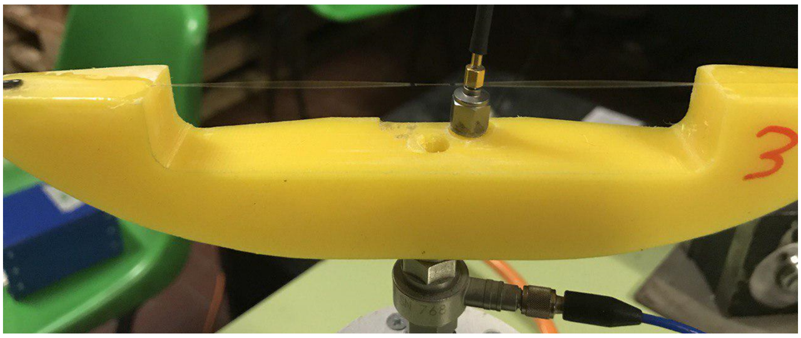


Fig. 2.11 Support 3 during testing

In any case, all the following graphs have been plotted considering only the first FFT peak, which corresponds to the exciting frequency in all the tests.

Three different testing sessions have been performed, varying frequencies at constant amplitudes of 1, 5 and 10 m/s^2, as shown in Fig. 2.13 and 2.14. It is clearly visible how the amplification peak is very close to the predicted natural frequency from the FEM modeling.

It can be seen how the amplification peaks tend to shift from the calculated natural frequency to higher or lower frequencies, depending on the support. This can be caused by several reasons: the material is not isotropic and thus not linear; the cyanoacrilate glue used to bond the fiber to the support exhibits a visco-plastic effect as shown in [44]; PLA and glue are sensitive to moisture and tests were performed

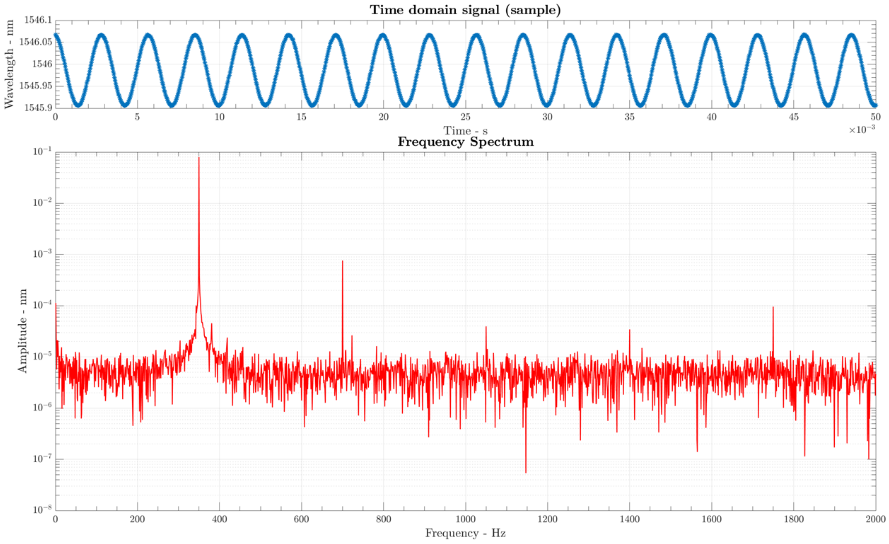


Fig. 2.12 Frequency response to a sinsuoid at 350 Hz

in different days without monitoring of the air humidity; tests at higher amplitudes might have caused a small plastic deformation in the plastic material. It wasn't possible to pinpoint the cause, or causes, leading to this phenomenon. Further testing will be needed with a increased number of samples.

It has to be noted that preliminary FEM analyses consider a much simplified model, consisting on the bare support, supposed to be made of an isotropic material with characteristics similar to the PLA used to create the supports. However, several strong assumptions have been made: firstly, on the material; in fact, 3D-printed plastics should not be considered as isotropic, given the process used to manufacture the component. However, considering a fill rate of 95% (i.e. how 'full' is the component), it can be considered as isotropic during this preliminary step. Furthermore, the presence of a glued, pre-tensioned glass fiber on the two extremes of the sensors should be considered during the FEM analyses as additional bending stiffness given the moderate preload amount; however, the cyanoacrilate glue is not very rigid and thus possess a moderate compliance, thus this effect doesn't seem to affect the results too much.

In fact, the experimental results match very closely the predicted resonance frequency even for this very simplified model, so it was decided not to perform a more in-depth FEM analysis as the data are consistent to what was observed during the tests.

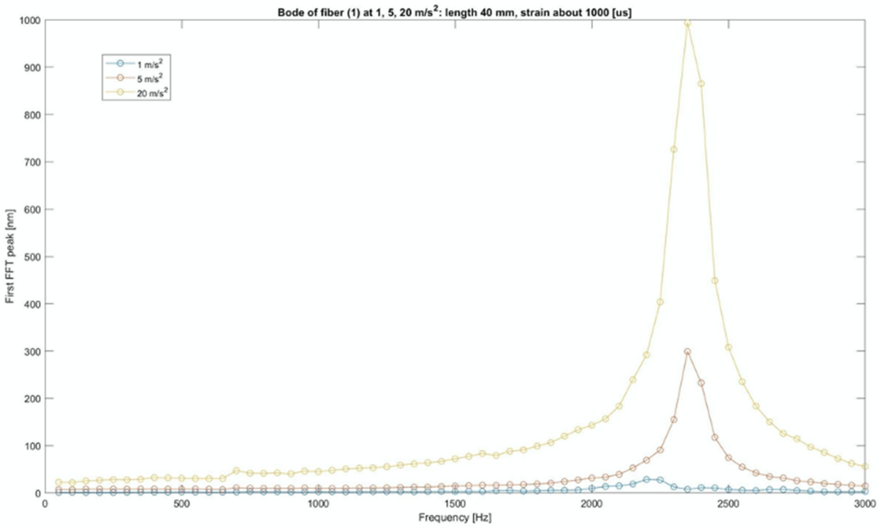
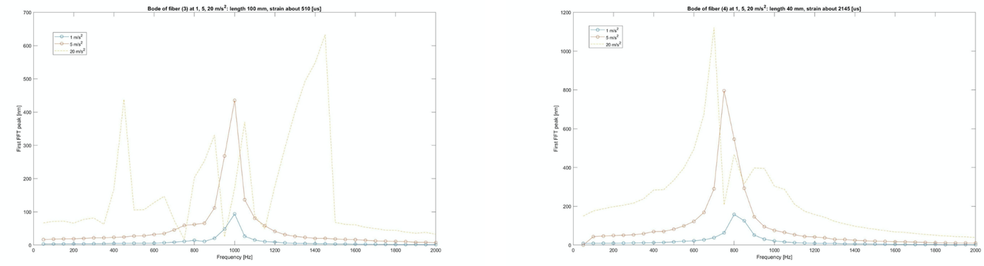


Fig. 2.13 Experimental Bode plot for combination 1-A



(a) Combination 4-B

(b) Combination 3-C

Fig. 2.14 Experimental Bode plot for combination 4-B and 3-C

Finally, in Table 2.2, the coefficient of amplification is reported for the different test performed, calculated as the ratio between the highest, first FFT peak amplitude with respect to the first 10 measurements, as to offer a baseline. It is visible how the amplification is not linear with amplitude, and tends to increase with higher values of it. For combination 4-B it wasn't possible to evaluate the coefficient of amplification for the test at 20 m/s^2, as something in the testing apparatus was influencing the measure. Finally, for combination 3-C, again for the test at 20 m/s^2, a characteristic curve was obtained but results are not very significative, as the maximum amplification is of about 4.4, which is an order of magnitude lower than the case at 5 m/s^2. This is probably due to some incorrect configuration in the testing equipment.

Overall, the design will need to be improved and made more resistant, as this proof of concept is very fragile having an exposed fiber. Additionally, some uncertainties will need to be removed by means off further testing with additional samples, and maybe developing more refined mathematical models. However, these preliminary tests show a good potential for selective frequency mechanical amplification for frequencies of interest, as the sensor design can be easily tailored as to have a natural frequency corresponding to the frequency of interest.

	A	B	C
1 m/s^2	14.7	15.1	23.2
5 m/s^2	30.6	44.8	41.3
20 m/s^2	33.5	-	4.4*

Table 2.2 Sensors amplification coefficients

2.4.2 Cantilever vibration sensor

Another solution explored during the PhD, is the cantilever. It is a simpler design that the previous, as it is composed of a simple extruded beam with prismatic section. Most of the theoretical design has been carried out by Dr. Antonio Marotta during his M.Sc. thesis project [45].

Considering the beam and frame of coordinates reported on the left of Figure 2.15, the force applied along the y-direction (right of the figure) will create a bending moment:

$$M_z = F_y \cdot x \tag{2.8}$$

From elastic line theory, one can obtain the differential relation between deflection y and bending moment M as such:

$$\frac{\partial^2 y}{\partial x^2} = \frac{M_z}{EI_z} \tag{2.9}$$

where E is the Young's modulus of the material, which is considered isotropic.

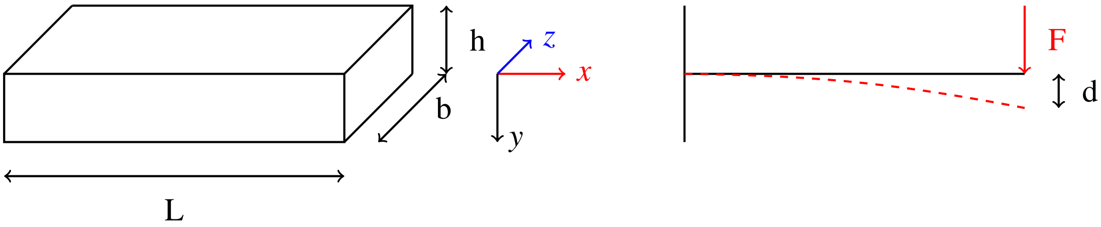


Fig. 2.15 Schematics of the cantilever beam

Solving the differential equation with fixed boundary conditions ($y(0) = 0$, $y'(0) = 0$) and with force applied at the free end, which is the case for the cantilever beam, one can obtain that:

$$d(x) = \frac{Fx^2}{6EI_z}(3L - x) \tag{2.10}$$

The maximum displacement can be found at $x = L$, and has a value of:

$$d_{max} = d(L) = \frac{FL^3}{3EI_z} \tag{2.11}$$

Stress can be derived from the Euler-Bernoulli beam theory, and can be expressed as:

$$\sigma_x(x,y) = \frac{M_y(x)}{I_z(x)}y = \frac{12Fx}{b(x)h^3}y \tag{2.12}$$

and by using the Hooke's law ($\sigma_x = E\varepsilon$), the deformation can be obtained:

$$\varepsilon_x(x,y) = \frac{12Fx}{b(x)h^3E}y \tag{2.13}$$

From the previous equation, it is possible to see that the strain changes along the length of the beam, even when considering a line at the same distance from the neutral axis. In our case, since the objective is to glue an FBG sensor along the two faces of the beam, this behavior is unwanted, since the FBG has a finite length and this could lead to wrongful measures and non-linear behavior of the sensing element.

To avoid this problem, one solution is to use an 'equal-deformation' beam, i.e. a beam where the deformation along the x-axis is constant, given a particular distance y from the neutral axis.

In order to do so, one can observe that:

$$\varepsilon_x(x,y) = \frac{12F}{Eh^3}\frac{x}{b(x)}y = k\frac{x}{b(x)}y \tag{2.14}$$

thus, to have constant strain along the axis, one must impose that:

$$\varepsilon_x(x,y) = k\frac{x}{b(x)}y \Rightarrow \frac{x}{b(x)} = \text{const.} = \frac{L}{b} \tag{2.15}$$

Applying the previous condition, i.e. a section reduction along the x-axis of the beam, the deformation is now only a function of y as such:

$$\varepsilon_x^*(y) = \frac{12FL}{Eh^3b}y \tag{2.16}$$

From the previous equation, it can be found that the maximum deformation along the whole beam is at the two faces, since their distance is the largest from the bending plane which in this case corresponds with the midplane of the beam. Numerically, these values are:

$$\varepsilon^*_{x,max} = \varepsilon^*_x(\pm\frac{h}{2}) = \pm\frac{6FL}{Eh^2b} \tag{2.17}$$

System dynamics

The vibrating system has been modeled as a second-order system, similar to a mass-damper oscillating system.

In fact, recalling Eqn. 2.11, the relation between force and maximum displacement can be expressed as:

$$F = \frac{3EI_z}{L^3} = Kd \tag{2.18}$$

where $K = \frac{3EI_z}{L^3}$ is defined as the flexural stiffness of the plate.

The second order differential equation (considering y as the degree of freedom) of a mass-spring system subjected to a periodic sinusoidal excitation and with viscous damping omitted, is:

$$m\ddot{y} + K\dot{y} = F\sin(\omega t) \tag{2.19}$$

where the usual solution applies:

$$y = \frac{F}{m(\omega_0^2 - \omega^2)} = \frac{a}{\omega_0^2 - \omega} \ , \ \omega_0^2 = \frac{K}{m} \tag{2.20}$$

where a is the acceleration and ω_0 is the natural frequency of the system, defined as the square root of the ratio between the flexural stiffness K and the mass m.

Now, by combining the previous equation with Eqns. 2.16 and 2.11, remembering that $I_z = \frac{1}{12}bh^3$, one can obtain:

$$\varepsilon = \frac{3d_{max} \cdot x}{L^2} = \frac{3y}{(\omega_0^2 - \omega^2)L^2} a \tag{2.21}$$

Considering the case when the excitation frequency is much smaller than the natural frequency of the system (i.e. $\omega << \omega_0$), the previous equation simplifies as:

$$\varepsilon = \frac{3y}{\omega_0^2 L^2} a \tag{2.22}$$

which is a relation that links the maximum deformation of a section of the beam with the external acceleration a.

The Bragg grating equation from the previous chapter is:

$$\frac{\Delta\lambda_B}{\lambda_B} = (1 + p_e)\Delta\varepsilon + (\alpha_\Lambda + \alpha_n)\Delta T \simeq (1 + p_e)\Delta\varepsilon \tag{2.23}$$

since the temperature component is considered negligible.

Since the fiber is glued to the upper (and lower) surface of the beam, it is reasonable to assume that:

$$\Delta\varepsilon = \alpha\varepsilon \tag{2.24}$$

where α is the glue transfer factor ($0 < \alpha < 1$). In other words, there must be some slippage introduced by the glue, which is taken into account with this particular factor.

Finally, applying this definition to Eqn. 2.21, the final relation between Bragg wavelength shift and acceleration can be obtained:

$$\frac{\Delta\lambda_B}{\lambda_B} = (1 + p_e)\frac{3\alpha y}{\omega_0^2 L^2} a \tag{2.25}$$

Final geometry and materials

After this preliminary analysis, a suitable material has to be chosen to manufacture the prototype. The choice for this initial run is rather limited, as the in-house production capability is limited to either 3d-printed PLA (Polylactic acid), a polymer, or milled aluminium alloy.

After a script has been used to obtain possible solutions, by fixing some dimensional parameters such as the various lengths. The imposed requirements are:

- ’Small’ size
- Resonant frequency above 200 Hz
- Maximum sensitivity
- Ease of manufacturing.

Sensitivity has been defined as the ratio of Bragg wavelength variation to unit of acceleration, which can be expressed as:

$$S = \frac{\Delta\lambda_B}{a} = (1+p_e)\frac{3\alpha y\lambda_B}{\omega_0^2 L^2} \tag{2.26}$$

In other words, the higher the sensitivity, the more the Bragg wavelength shifts for a given acceleration value; in other words, one can obtain a finer measure by having a higher sensitivity.

Considering all the previous conditions, the same design has been evaluated using both PLA and aluminium, obtaining the following values:

Material	K [N/m]	L [mm]	B [mm]	h [mm]	m [g]	ω [Hz]	S [m/g]
Al	1.9e4	10	1.5	1	11.6	206.86	1.35e-11
PLA	9.8e3	10	1.5	1	0.58	206.52	1.35e-11

Table 2.3 Sample design comparison

The previous table shows that the excitation mass *m* is more than an order of magnitude greater for the aluminium compared to PLA; in particular, this sensor dimensions are really small and this could lead to manufacturing and volume problems when including a mass this big.

After considering several designs, the choice was made on the following parameters set:

L [mm]	b [mm]	h [mm]	m [g]	ω [Hz]	S [m/g]
28	6	4	5	232.5	4.4e-12

Table 2.4 Final prototype dimension

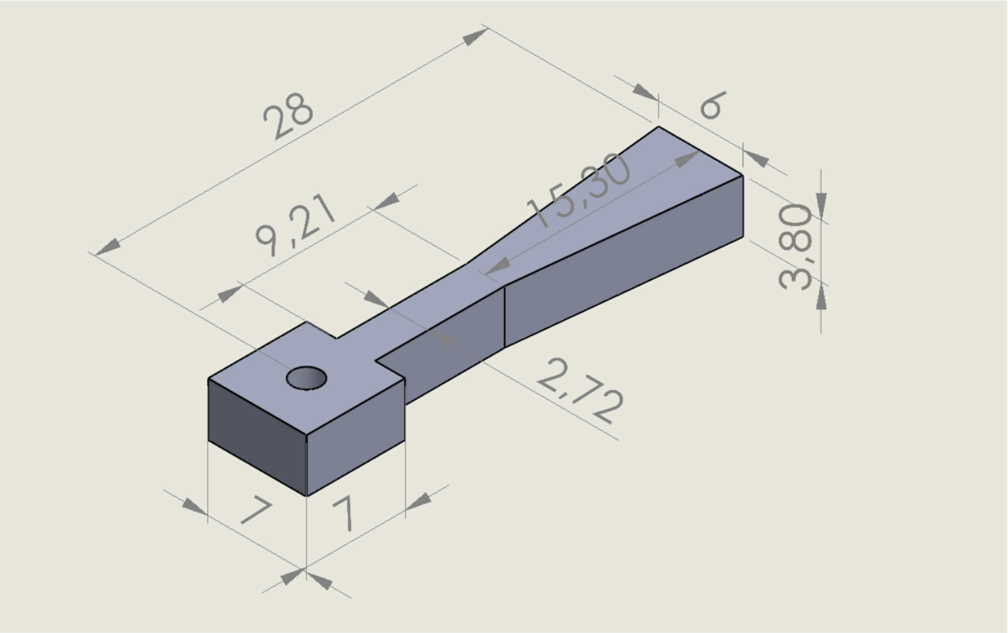


Fig. 2.16 Modified beam FEM sizes (in [mm])

However, this simplified geometry has several problems regarding manufacturing, since the free edge of the beam is only 1 mm wide, which poses a challenge for gluing the excitation masses; in fact, the chosen mass is 5 g, which is more than two big lead fishing weights.

To amend this problem the geometry has been slightly altered to maintain sensitivity and resonant frequency values from the original design. The final, modified design has a square holder for the weights at the free end, and has a lower overall thickness. Additionally, the exciting mass has been reduced to 4 g, since this allows the use of two single lead fishing weights. The last modification is a necking down of the beam section towards the holder, since that terminal portion of the beam does not provide support for the fiber. Final sizes are shown in Figure 2.16.

After final sizing, some FEM analyses have been performed to validate the analytical results. In Figure 2.17, the total strain of the sensing element is shown, from a linear analysis. As forecast by the theory, the strain value is constant along the beam for a given value of y.

However, as explained previously, the necking down to a constant section changes this behavior and can be observed in the terminal region of the beam.

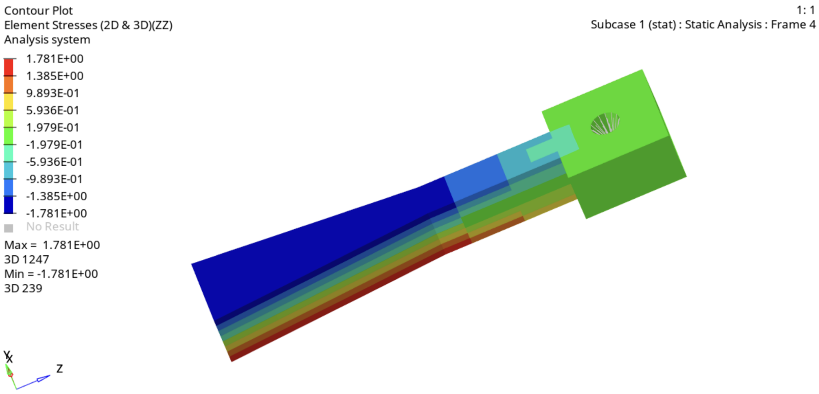


Fig. 2.17 Modified beam FEM strain results

Furthermore, a modal analysis has been carried out and is shown in Figure 2.18, where the natural frequency is predicted as 235.6 Hz, a value very close to the theoretical value.

To set-up the FEM analysis, Hypermesh 2021 has been used. The type of elements used are HEXA-type solid elements, initially meshed on the top surface and then extruded acorss the thickness of the object. All the nodes on the base face have been constrained in all six degrees of freedom to simulate the fixed joint, as that face is bonded to the rest of the support. Finally, the excitation mass has been modeled using a single node in the center of the end cavity. The node has been connected to all nodes of the cylindrical opening with rigid beam elements, RBE2.

Finally, in Figure 2.19a, the holder for the cantilever beam is shown. It is a simple rectangular holder designed to maintain the beams in a fixed position and stop them from moving along the lateral axis. In Figure 2.19b the base support for

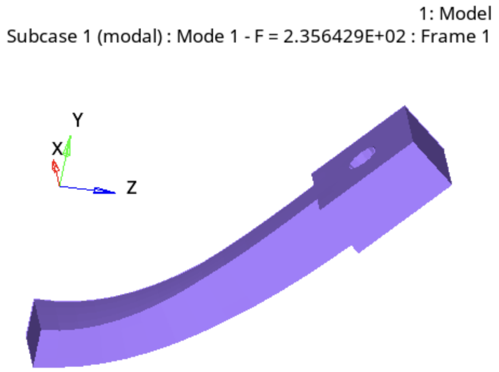
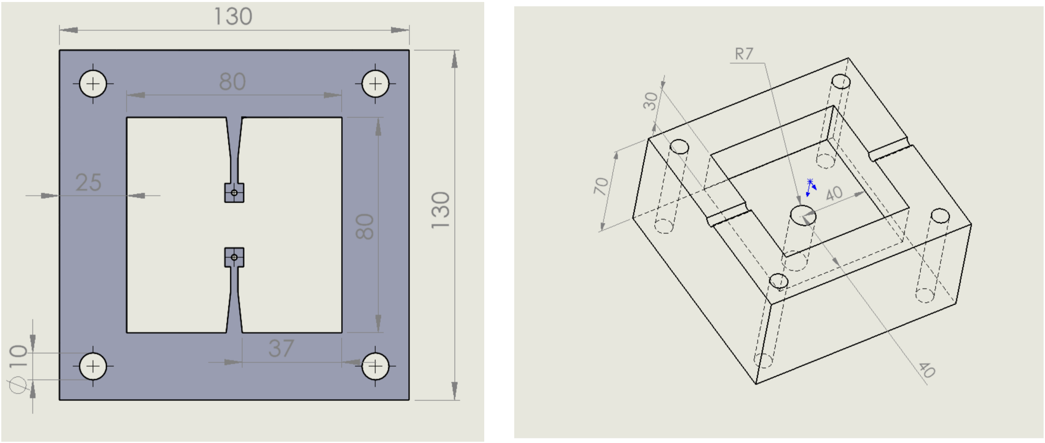


Fig. 2.18 Modified beam natural frequency (FEM)



(a) Sensor mid support

(b) Sensor base support

Fig. 2.19 Final sensor design with supports

the whole structure is shown; the need arises from making sure that, during vibration, the two cantilever beams do not slam against a possible lower reference surface, thus providing protection to the sensor. Additionally, the base support has 4 holes in the corner for fasteners, while the hole in the center bottom is used to secure the whole sensor to the shaker, as will be shown in the next section, to perform validation tests.

Preliminary testing

After printing the sensor, and bonding the the optical fibers using cyanoacrilate glue, an experimental setup has been arranged to perform several vibration tests of the sensor.

Setup is shown in the Figures 2.20 and 2.21. In Fig. 2.20a, part of the experimental setup is set on a very heavy steel table, as to reduce the effect of spurious environmental vibrations. The sensor is mounted on the shaker, and the coated optical fibers (in red) are connected to the Smart Fibres SmartScan FBG interrogator, which can sense the strain of the FBGs which are mounted on the cantilever.

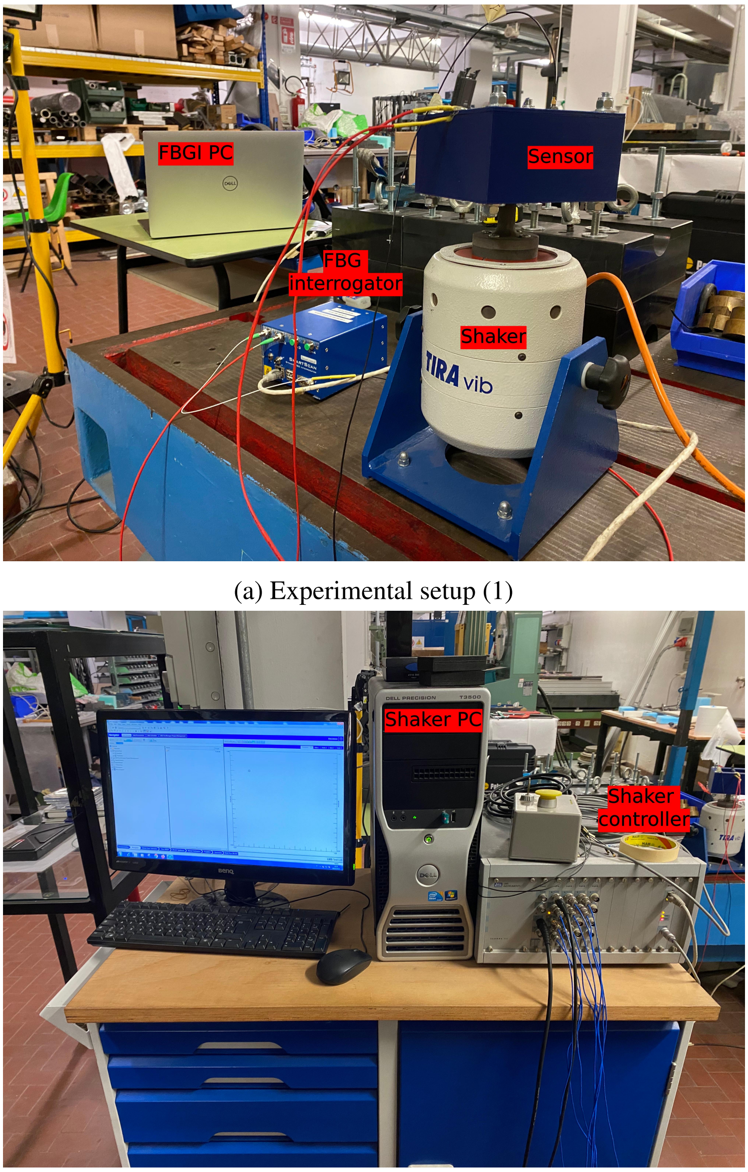
Regarding said sensors, it was planned to have two identical gratings (i.e. with the same Bragg wavelength) on a single cantilever, since this was a preliminary test. However, after gluing and curing of the glue, it was found out that, while the sensor on the top face of the cantilever was in fact centered at ca. 1538 nm, the one glued to the bottom face had a Bragg wavelength of ca. 1530 nm, which is unfortunate since it is very close to the lower detection edge of the interrogator, which is 1528 nm.

The PC described as 'FBGI PC' is used to run the SmartScan software for acquisition of the data, with a sample rate set to 2.5 kHz, which is plenty since the range of interest for mechanical vibrations are set to 200 Hz as a design limit. Another view of the software is shown in Fig. 2.21b.

In Figure 2.20b, the PC used to generate the excitation vibration is shown, along with the hardware controller for the shaker. The PC generates a sinusoidal voltage profile, which is then amplified by the amplifier shown in Fig. 2.21c, to provide a strong signal to drive the shaker.

Finally, in Figure 2.21a, a detailed view of the sensing element is shown. The sensing cantilever is the one on the left hand side, while the right cantilever has no fibers attached to it, since this was a preliminary test. The fibers have a protective plastic coating (in yellow) up to the interface with the body of the support. Then, only the bare fiber is set along the length of the cantilever. Hot glue has been used to secure the plastic protective casing to the support. Finally, an accelerometer has been installed on the cantilever with no fibers to provide a visual feedback on the 'Shaker PC' to check if the system was behaving as expected.

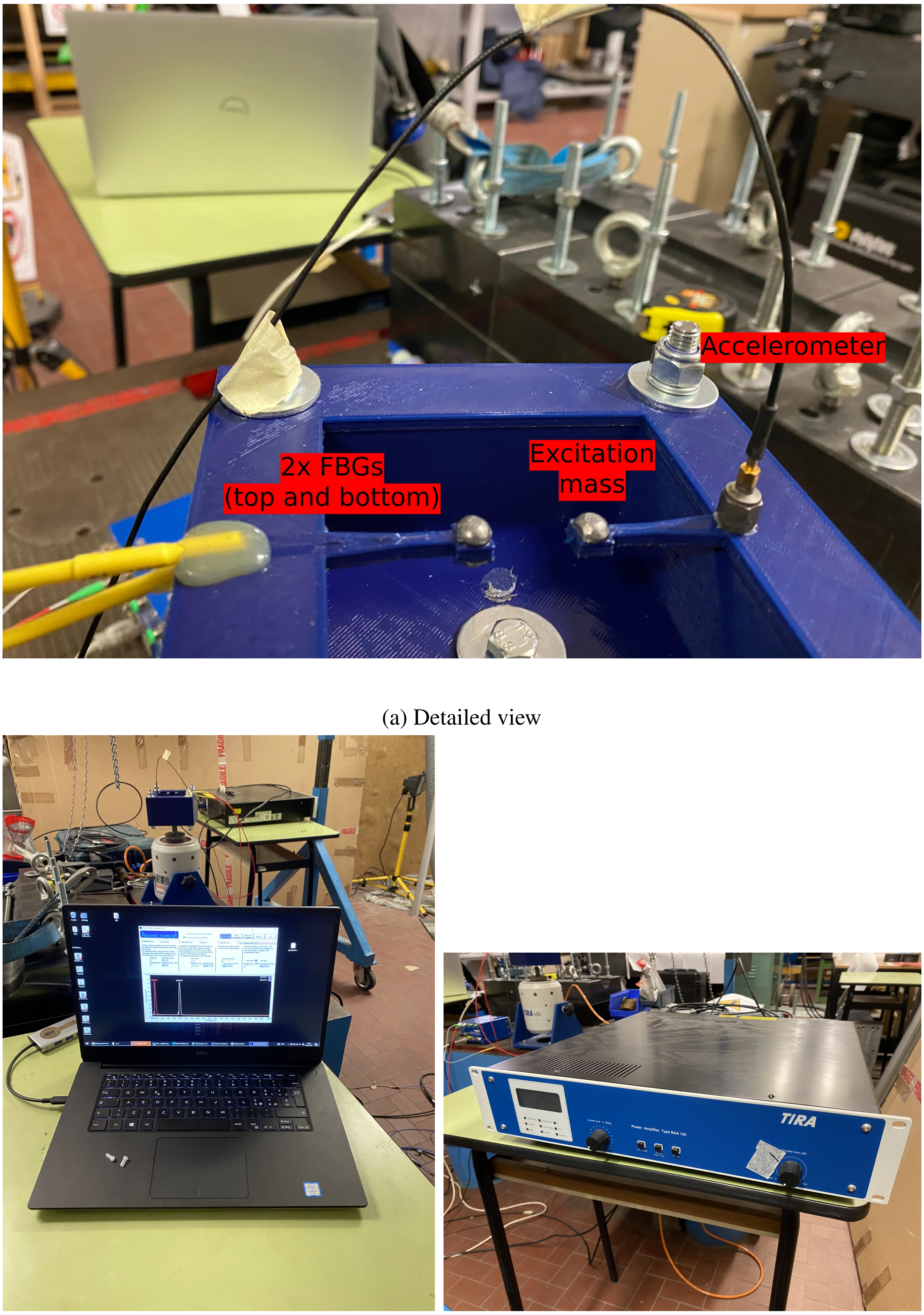
It has to be noted that the tests were performed in open loop conditions, as it was not possible to create a feedback loop using the accelerometer which was used, as previously stated, as simple visual indicator. Thus, the controlled variable was the driving voltage (which was kept constant, while frequency was swept) and not the amplitude of the vibrations.



(a) Experimental setup (1)

(b) Experimental setup (2)

Fig. 2.20 Experimental setup overview



(a) Detailed view

(b) Interrogation PC

(c) Amplifier

Fig. 2.21 Experimental setup overview (cont.)

Test results

Several simple tests have been performed to experimentally validate the theoretical results. Initially, constant-amplitude constant-frequency tests have been performed to characterize the system. Since the time-domain signals are not very descriptive, the FFT of all trials is shown in Figure 2.22. All signals are plotted on the same graph since all tests have same voltage amplitude (input) and length, thus the FFT signals are directly comparable.

As previously stated, it was not possible to directly control excitation amplitude, thus a constant voltage of 0.570 V was set for all trials. The value has been chosen as to have an average response amplitude of 1 *g* of acceleration at 5 Hz.

The response peaks are clearly visible for each of the trials as sharp peaks in the FFT, and it can be observed that the amplitude initially increases up to the 300 Hz test, and then start decreasing for the following tests.

However, to find the natural frequency of the sensor, a frequency sweep has been also performed. The continuous sweep is not a perfect test since the inertial contribute is not completely removed. However, given the very modest mass of the sensor, and the rather long test, it is still a useful evaluation.

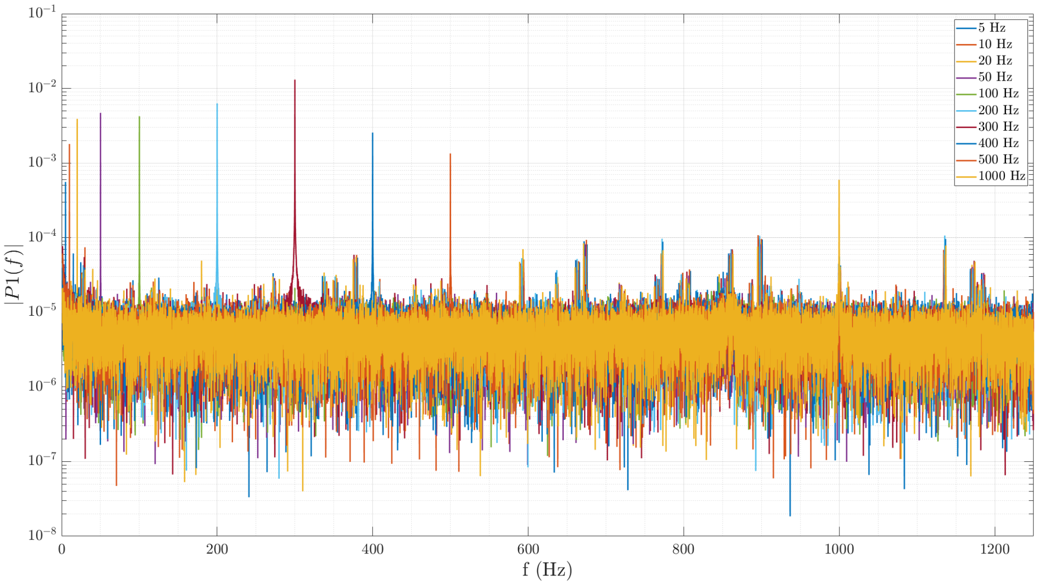


Fig. 2.22 FFT of all constant amplitude tests

In Figure 2.23, the time response of both sensor is shown (sensor 1 is on the top face, sensor 2 on the bottom). The first ca. 4 s are used to calibrate the normalization performed on the signal. In fact, the figure shows the variation of Bragg wavelength of each sensor (plotted in picometers) from the mean value of the first 7500 samples (ca. 3 s), to obtain a more accurate measure rather than using the stated Bragg wavelength, since the nominal value can be influenced by factors such as glueing and involuntary pre-tensioning.

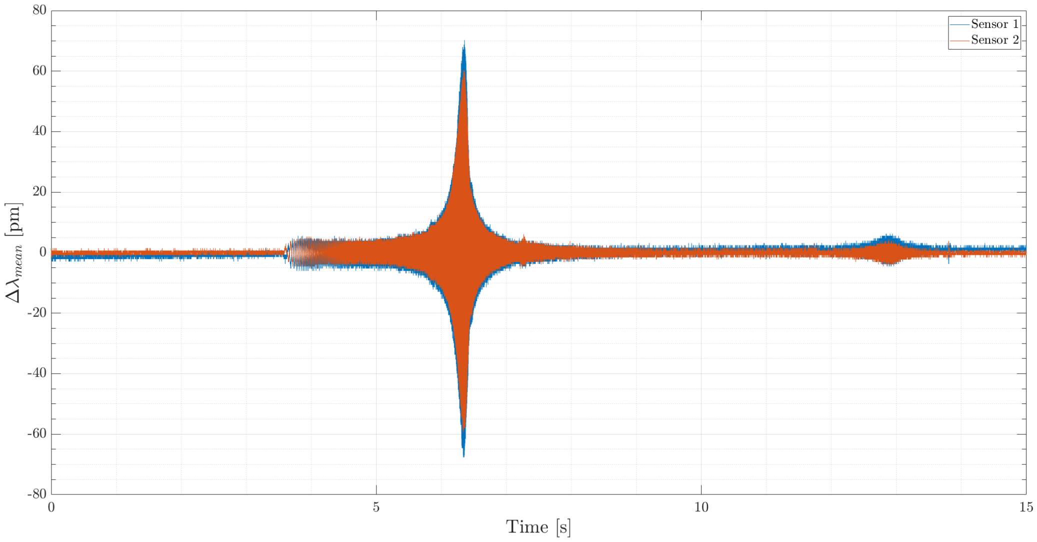


Fig. 2.23 Time response of frequency sweep (0-1000 Hz)

As for the previous case, the FFT of the signal has been evaluated and is shown in Figure 2.24. In this case, the resonant frequency is clearly visible, and it is at a frequency of $\sim$ 273 Hz, which is not too dissimilar from the one predicted by the FEM analysis, which was 235.6 Hz.

The discrepancy can be attributed to several causes: firstly, the incorrect modeling of the material. In fact, generic values for the material (PLA) properties have been chosen, which are surely different from the one of the filament used. Furthermore, the material has been considered isotropic, which is not a good assumptions for 3D-printed components, as the z-axis (extrusion axis) has markedly different properties from the x and y-axis.

Moreover, the component does not have a 100% fillrate, which means that the component is partly empty inside, thus changing the mass/stiffness ratio; this phenomenon is limited to the lower support and not the medium support, which is the one including the cantilevers and has a 100% fillrate. In any case, the FEM simulations provide a good ballpark estimate for the resonant frequency.

Referring to Fig. 2.24, the amplification is circa constant in the range of interest ($0-200$ Hz). However, for very low frequencies ($f < 15$ Hz), a significant attenuation is present. The cause is unclear and further testing is needed to determinate the cause. One hypothesis is that the structure damping has a relevant effect at such low frequencies.

Another amplification peak can be found at around 914 Hz. This amplitude does not correspond to any harmonic of the sensor, and thus is probably a resonance frequency of the lower support that manages to excite the cantilever. However, it has to be noted that this frequency is far beyond the working range of the sensor.

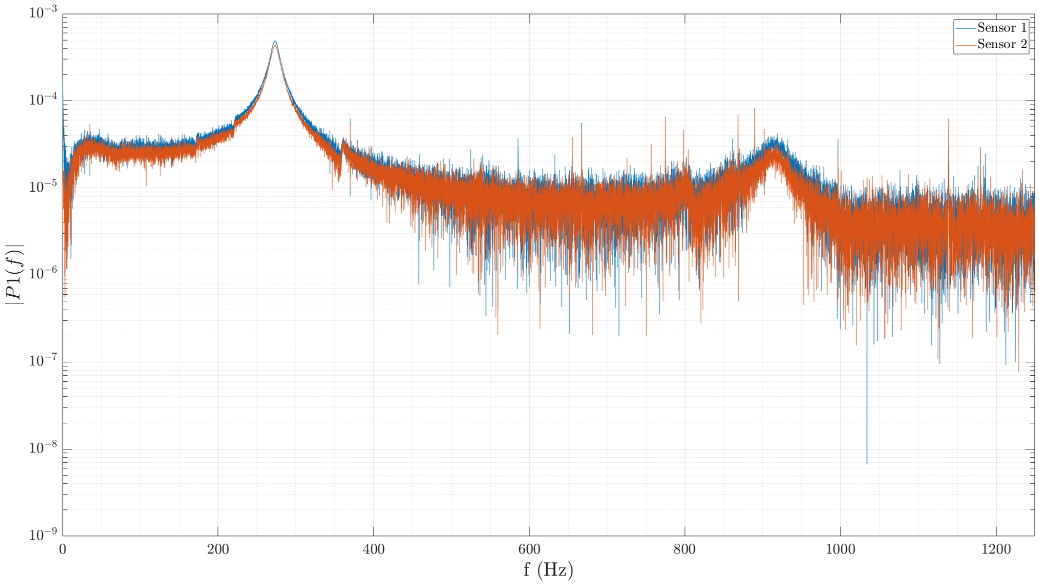


Fig. 2.24 FFT of frequency sweep (0-1000 Hz)

Additional sweeps have been performed at different amplitudes and limited to 350 Hz, as frequency above that are not of interest for this particular sensor. The FFT are shown in Figure 2.25, where a voltage amplitude of 1 V was used, and in Figure 2.26, where 2 V were set. Once again, the amplitude response is quite linear in the region of interest, and a strong attenuation can be observed for low frequencies.

It is also visible how the second harmonic ($\sim$ 546 Hz) is also excited at higher amplitudes, and the higher the excitation amplitude, the more the second harmonic is excited.

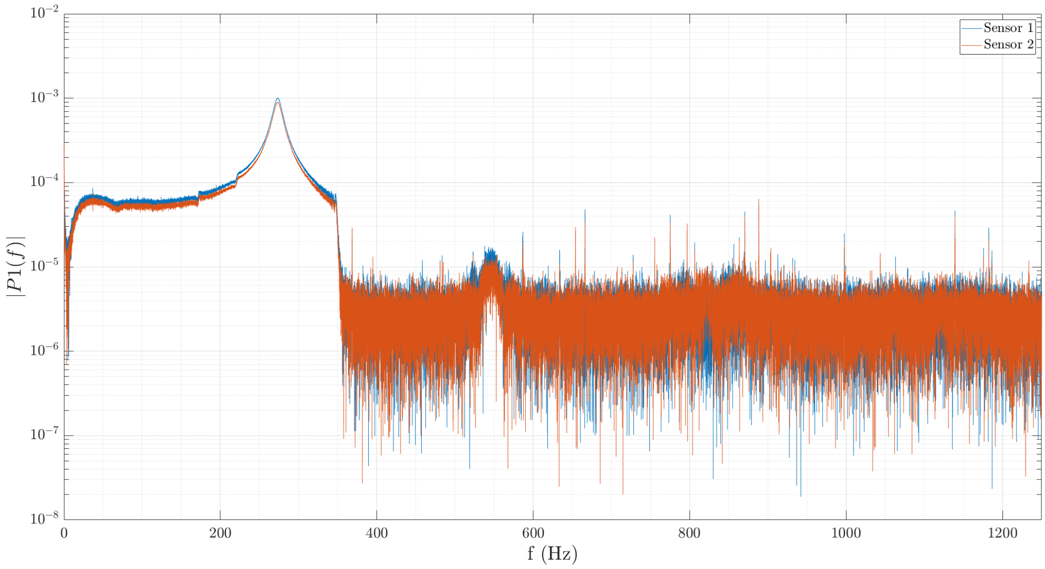


Fig. 2.25 FFT of frequency sweep, 1 V (0-350 Hz)

However, in Fig. 2.26, the FFT of sensor 2 has a very strange response and the power spectral density also three or more order of magnitudes greater than the one of sensor 1. This could be easily explained by looking at the time-domain signal of the reflected wavelength, shown in Figure 2.27. It is evident that for large part of the test, readings from sensor 2 go out of bounds. This happens for two reasons: firstly, because the Bragg wavelength is very close to the lower edge of the measuring window as previously explained, and secondly since the amplitude is larger compared to previous test.

Thus, in the time domain, there is a continous cycle between a value very close to the Bragg wavelength and zero, which means that the reflected wavelength goes below the minimum wavelength readable by the interrogator. This phenomenon can be easily avoided by choosing sensors whose Bragg wavelength is in the middle of the measuring window, i.e. around 1550 nm in the case of SmartScan interrogator.

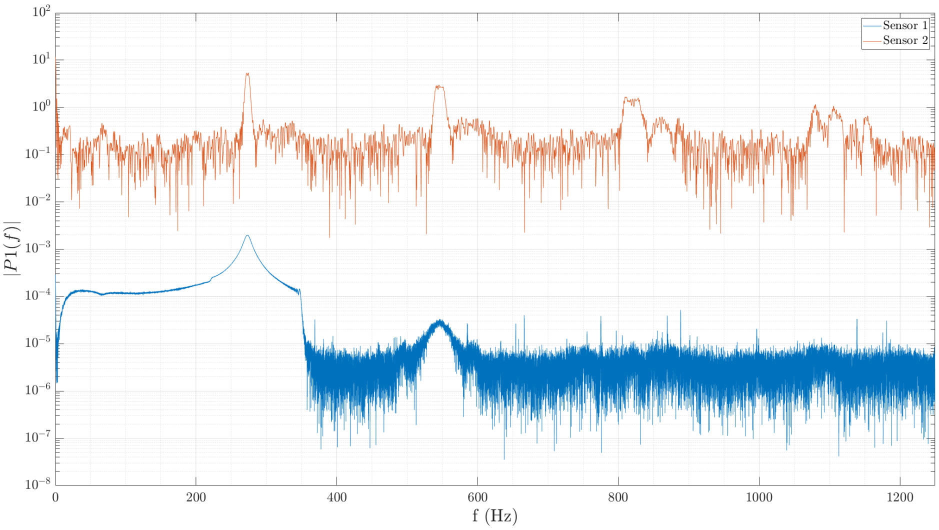


Fig. 2.26 FFT of frequency sweep, 2 V (0-350 Hz)

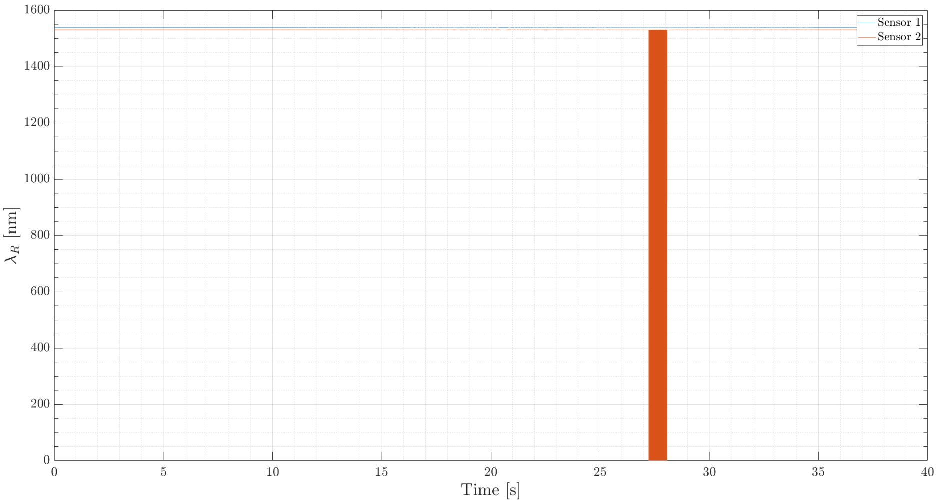


Fig. 2.27 Time domain response of frequency sweep, 2 V (0-350 Hz)

Experimental sensitivity analysis

An experimental sensitivity analysis has also been carried out thanks to the data provided by the accelerometer placed on top of the support in a position as close as possible to the cantilever beam. It has to be noted that it has been placed on the opposite side where the fibers are placed, given the constraints posed by the fibers themselves and the drop of hot glue needed to secure them to the top of the support.

Acceleration data are collected for all the different tests performed, both for constant sinsuoidal excitation and also for constant-amplitude chirps, which are useful for structure characterization.

One example of accelerometric data is shown in Fig. 2.28, where the sine excitation is clearly visible.

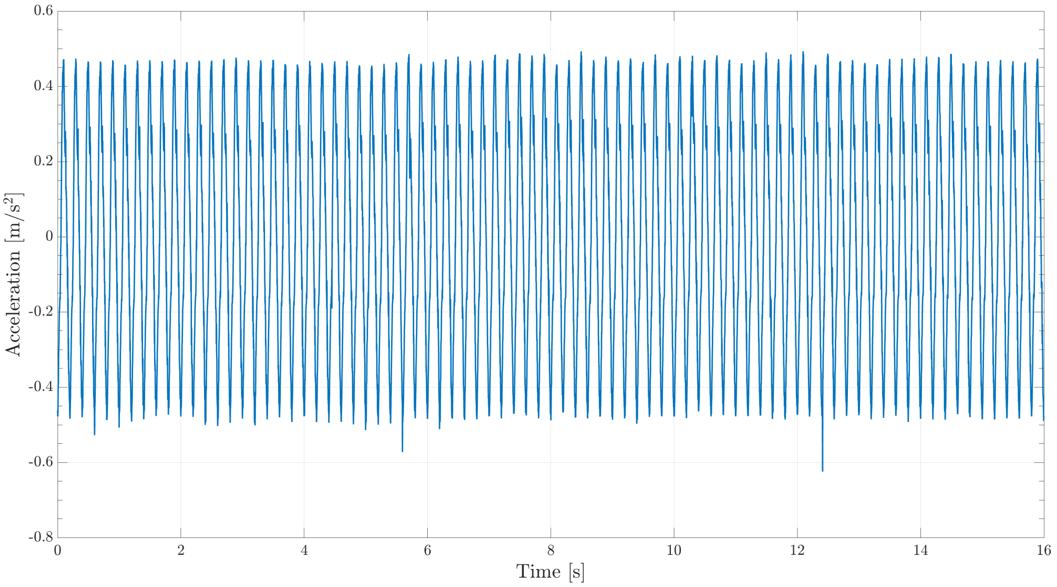


Fig. 2.28 Acceleration curve as read by accelerometer, 5 Hz

On the other hand, the chirp response is shown in Fig. 2.29, which is the FFT of a constant-amplitude chirp between 0 and 350 Hz, and is the same test as the one shown in Fig. 2.25. The first support natural frequency is easily detectable at 220 Hz, and also the first harmonics at 440 Hz, even tough that frequency is not directly excited. A minor resonance is also observable at 170 Hz and will need further investigation. It has to be noted that in the range 15-200 Hz the response is almost constant, which is useful since the objective is to characterize the cantilever reducing as much as possible the effect of the support.

Referring again to Fig. 2.25, the effect of the support dynamics is very modest; in fact, the two resonances at 170 Hz and 220 Hz can be detected but do not provide significant alteration of the FBGs readings.

All that being said, in Table 2.5, the experimental sensitivity data are obtained. By simply dividing the amplitude read by the FBG by the excitation amplitude read

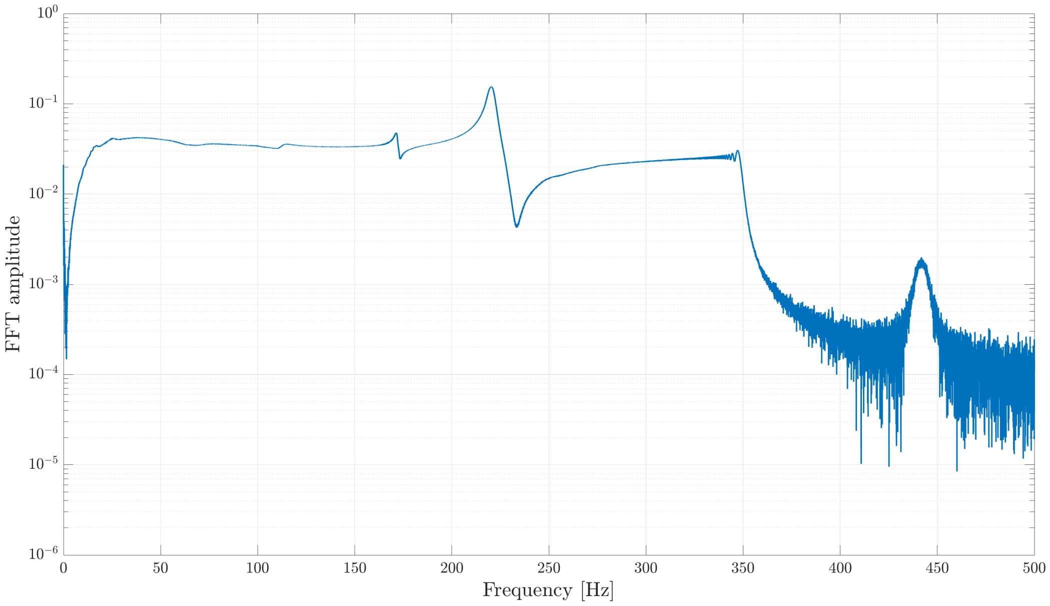


Fig. 2.29 Support dynamic response, 0-350 Hz chirp, 1 V

by the accelerometer, the sensitivity can be obtained. For all curves, peak-to-peak measurements are used.

For each measure, the minimum detectable acceleration is also calculated, by setting the Bragg wavelength variation as the minimum, repeatable measure that the FBGI can elaborate, which is 1 pm.

It can be observed that the sensitivity is almost constant in the range of interest, i.e. for frequencies lower than 200 Hz. On the other hand, when the excitation frequency approaches the natural frequency, a sharp increase in sensitivity can be observed. This behavior can be easily explained by the natural amplification provided by the cantilever which greatly increases the oscillation amplitude and thus the strain applied to the FBGs. Sensitivity data are also graphed in Fig. 2.30.

This is a preliminary series of test and other testing campaigns will need to be performed to completely characterize the effect of the support and the response curve of the cantilever. In fact, the experimental values are quite higher from the theoretical ones and the cause will need to be investigated.

Frequency [Hz]	Excitation amplitude [m/s^2]	FBG amplitude [pm]	Sensitivity [pm/(m/s^2)]	1-pm acc. [m/s^2]	Sensitivity [pm/g]
5	0,952	1,5	1,58	0,63	15,46
10	2,91	5	1,72	0,58	16,86
20	5,95	6,5	1,09	0,92	10,72
50	6,87	8,2	1,19	0,84	11,71
100	5,55	8	1,44	0,69	14,14
200	4,21	11	2,61	0,38	25,63
300	3,79	24	6,33	0,16	62,12
400	3,43	5	1,46	0,69	14,30
500	3,07	3	0,98	1,02	9,59
1000	9,76	2	0,20	4,88	2,01

Table 2.5 Experimental sensitivity data

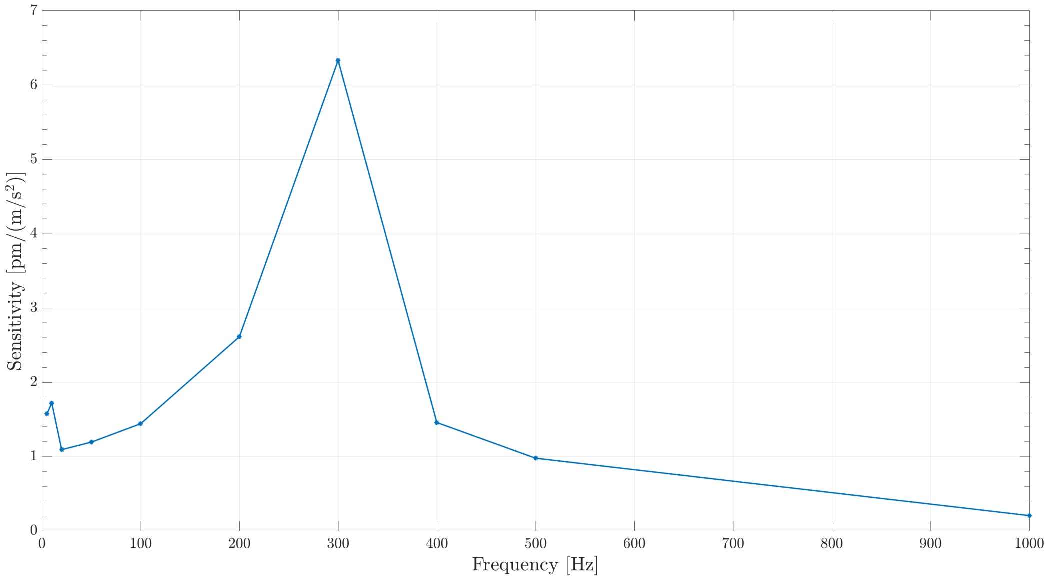
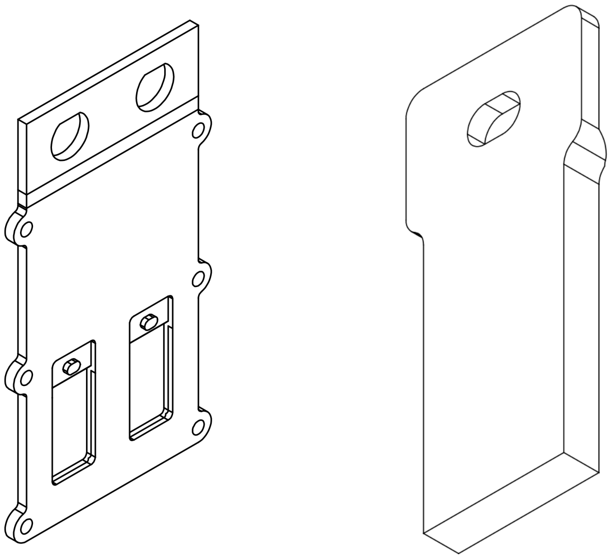


Fig. 2.30 Experimental sensitivity curve

2.4.3 Compensated thermal sensor

Another sensor which was conceived was a combined strain-thermal sensor, again based on FBG sensors. Due to lack of time, besides the numerical modeling, a limited test was carried out.

The main design philosophy is to have redundant measures, which is accomplished by using more than a single sensing element to avoid having single point of failure. The general design is shown in Figure 2.31, and is composed of an aluminum support and two laminae, one again made of aluminum alloy 6082 and the other made of low carbon steel C45.



(a) Support

(b) Lamina

Fig. 2.31 Sensor components schematics

The objective of this sensor is to measure both temperature and strain applied to the sensor, since FBGs are sensitive to both mechanical and thermal-induced strains.

In the complete version of the sensor, one additional FBG sensor will be bonded to the support, as to be also sensitive to mechanical strain. The following analysis refers to one sensor bonded to the support and one to the aluminum lamina.

Two different materials are used for the laminae as to also allow testing of the interaction between glue, coating, fiber and metal, which will be the scope of future works.

Numerical model

The basic FBG equation relating strain, temperature and wavelength is the usual Eq. 2.7, which once again is:

$$\frac{\Delta\lambda_B}{\lambda_B} = (1 - p_e)\varepsilon + (\alpha_\Lambda + \alpha_n)\Delta T \tag{2.27}$$

However, when bonded to another material, and especially if the support mass is much greater than that of the fiber, the effects of said material become the driving effect on the FBG readings.

In fact, for a generic isotropic material, strain can be induced by both mechanical effects and temperature variations, which are described by the following equations:

$$\varepsilon_M = \frac{\sigma}{E} = \frac{F}{A \cdot E} \tag{2.28}$$

$$\varepsilon_T = \alpha\Delta T \tag{2.29}$$

where F is the force applied to the specimen, σ is stress, A is the sectional area, E is the Young modulus of the material and α is the linear coefficient of thermal expansion. The assumption is that the specimen is only loaded in axial direction by traction or compression.

Considering the FBG which is bonded to the support, it will be subject to both thermal and mechanical strains, thus the normalized wavelength shift can be expressed as:

$$\frac{\Delta\lambda_1}{\lambda_1} = k'_{\varepsilon,1}\left(\frac{F}{AE_{al}} + \alpha_{al}\Delta T\right) + k_{T,fiber}\Delta T \tag{2.30}$$

where $k'_{\varepsilon,1}$ is the modified k_ε considering the effect of the bonding, which is harder to model better so this model relies on an experimental characterization to obtain the numerical values.

On the other hand, the fiber bonded to the lamina will only be subject to thermal-induced material strain, thus the normalized wavelength shift is:

$$\frac{\Delta\lambda_2}{\lambda_2} = k'_{\varepsilon,2}\alpha_{al}\Delta T + k_{T,fiber}\Delta T = k_{T2,eq}\Delta T \tag{2.31}$$

Now, subtracting the two previous equation, we can obtain:

$$\frac{\Delta\lambda_1}{\lambda_1} - \frac{\Delta\lambda_2}{\lambda_2} = k'_{\varepsilon,1}\frac{F}{AE_{al}} + (k'_{\varepsilon,1} - k'_{\varepsilon,2})\alpha_{al}\Delta T \tag{2.32}$$

The modified strain coefficients are defined as:

$$k'_{\varepsilon,j} = k_{B_j}k_\varepsilon \tag{2.33}$$

where k_B is a bonding constant which encapsulate the effect of the glue on the strain transfer, while $k_\varepsilon = 1 - p_E$ is the nominal strain coefficient of the fiber.

Substituting the previous equation in Eqn. 2.32, after some algebric passages one can obtain:

$$\frac{\Delta\lambda_1}{\lambda_1} - \frac{\Delta\lambda_2}{\lambda_2} = (1 - p_E)\left(k_{B,1}\frac{F}{E_{al}A} + (k_{B,1} - k_{B,2})\alpha_{al}\Delta T\right) \tag{2.34}$$

where it was assumed that both fibers are identical, thus $p_{E,1} = p_{E,2}$ and also that the thermal-induced strain component directly on the fiber is equal, i.e. $k_{T,fiber}$ is the same for both FBGs. Additionally, it is assumed that thermal equilibrium has been reached.

Observing the previous equation, it is possible to see that the thermal and mechanical components are not directly separable in a realistic case, since the magnitude of the bonding effect is unclear. However, if the difference between the two FBGs bonds is small, then the difference between the normalized wavelengths shift is almost exclusively due to the mechanical-induced strain.

Additionally, several parameters are unknown and will need to be experimentally characterized, starting from the two bonding constants $k_{B_{1,2}}$.

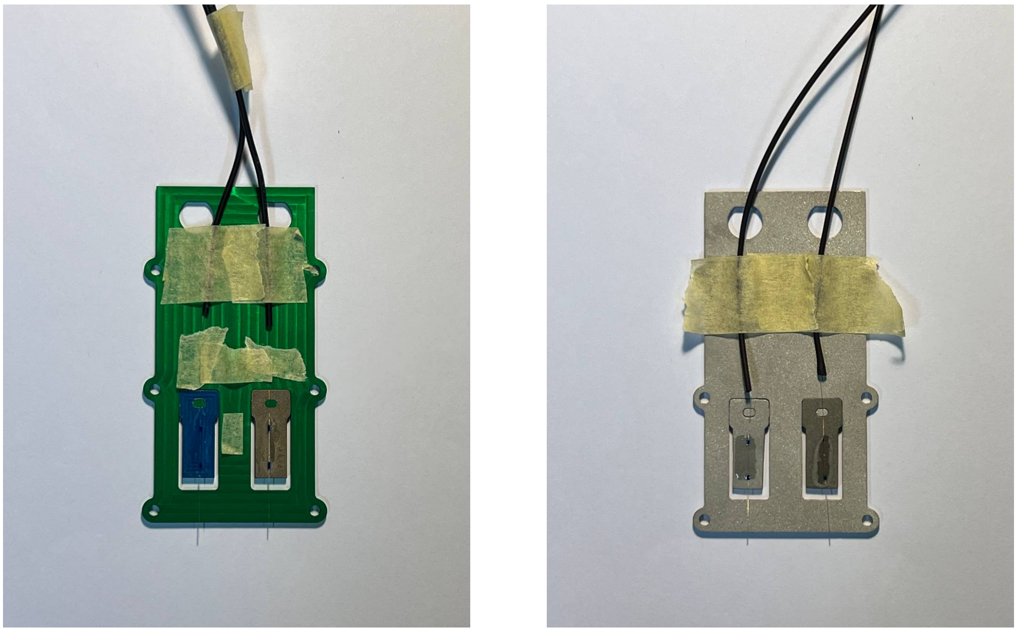
In any case, if only the temperature measure is needed, the laminae bonded sensor can be used by itself. As reported in Equation 2.31, that sensor can be thought as being purely proportional to temperature only. This stems from the fact that no mechanical constraints limit the free expansion of the lamina and thus of the fiber. To do so, a calibration will be needed to obtain the value of the equivalent thermal coefficient for the lamina-FBG assembly, which is denoted with $k_{T,eq}$ in Eqn. 2.31.

Experimental determination of k_B

Due to time constraints, a limited number of tests has been carried out. In this test, the focus was on the determination of the bonding coefficient k_B, limited to the case of purely thermal strain.

To do so, two physical samples have been manufactured, with the same specifics, and are shown in Fig. 2.32. The anodized specimen has two acrylate-coated FBGs bonded to each of the laminae, while the non-anodized specimen uses polyammidic-coated fibers, with the FBG element exposed. All the fibers have been bonded to the metal supports using cyanoacrilate resin.

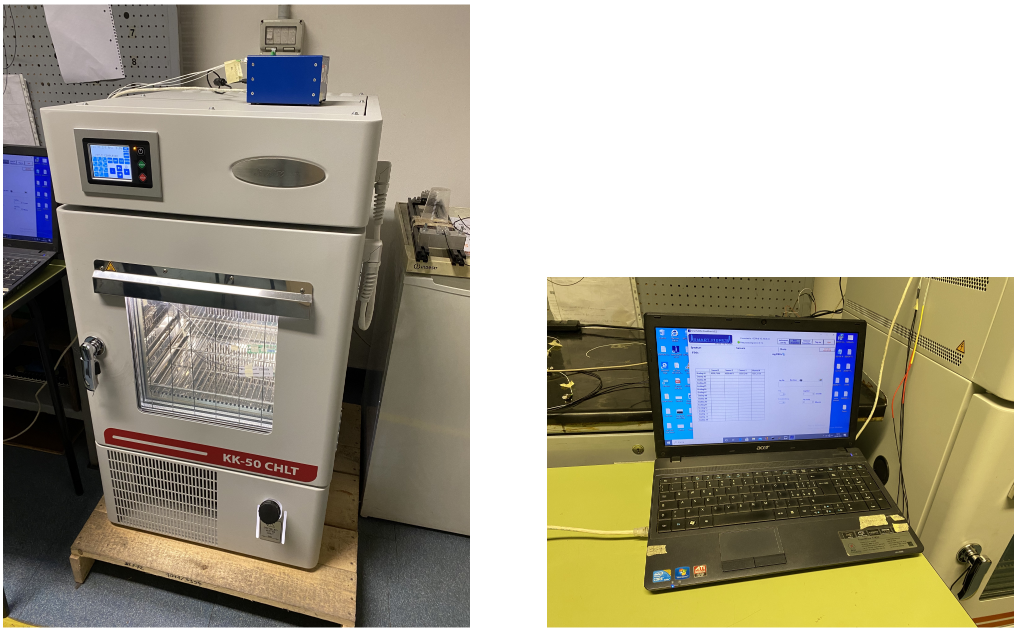
Both specimens have been inserted inside a Beger KK-50 climatic chamber (Fig. 2.33a), used to precisely control temperature setpoints. Humidity has not been changed during the test and was kept circa constant at about 50%. The thermal profile was set as a temperature stair, with 10 °C increments, with each step lasting for 30 minutes. The FBGs wavelengths are analyzed by the usual Smart Fibres Smartscan interrogator (Fig. 2.33a, on top of the climatic chamber), while the acquisition is carried out using the laptop shown in Fig. 2.33b, which is also used to monitor and log the chamber setpoint and actual temperature (Fig. 2.34).



(a) Specimen with coated sensors

(b) Specimen with uncoated sensors

Fig. 2.32 Supports for testing k_B



(a) Climatic chamber and FBGI

(b) Acquisition laptop

Fig. 2.33 Testing apparatus

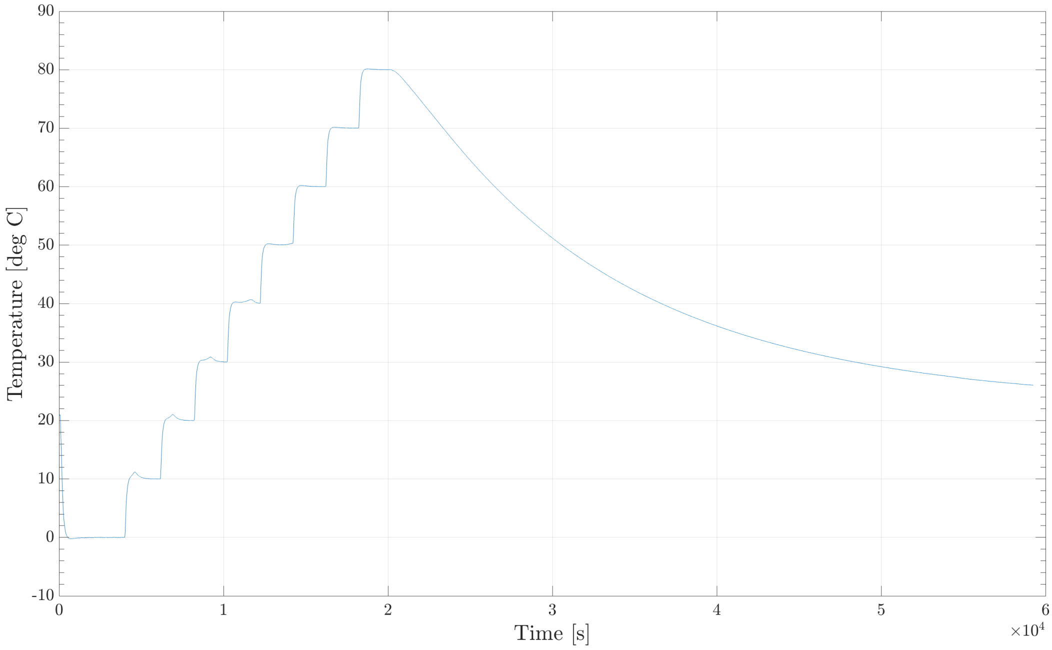


Fig. 2.34 Chamber temperature graph

The raw readings from the FBG interrogator (FBGI) are show in Fig. 2.35. While the acrylate-coated FBGs exhibits a standard behavior, mirroring the temperature setpoints of the chamber (including the overshoots visible especially at 10, 20 and 30 °C), the uncoated variants show a strange behavior at higher temperatures. It seems that the cyanoacrilate does not bond correctly either to the bare glass, the polyammidic coating, or both. It is possible that the fiber starts slipping and thus provides a reading which is not constant. Thus, viscous creep due to poor bonding could explain the behavior of the uncoated fiber.

The raw data have been used to extract the equilibrium wavelength values, which was not really possible for the naked fibers. In that case, the last point of each temperature step has been chosen to maintain consistency; however, new measurements will be needed to better characterize this phenomenon.

Nonetheless, the data points have been linearly interpolated, and data are shown in Fig. 2.36 and Fig. 2.37.

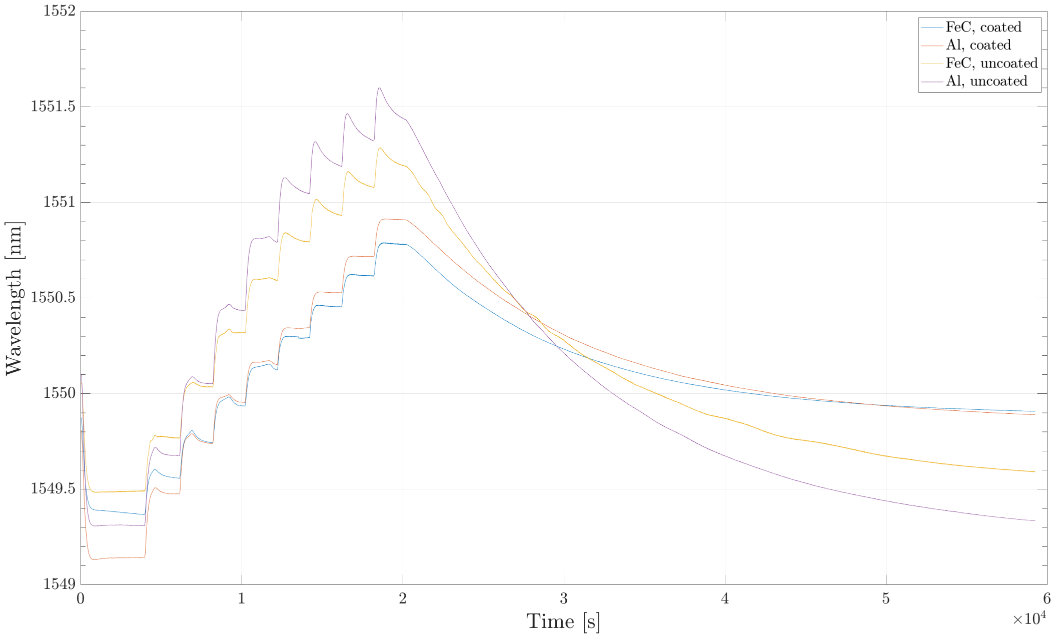


Fig. 2.35 Readings from the FBGI

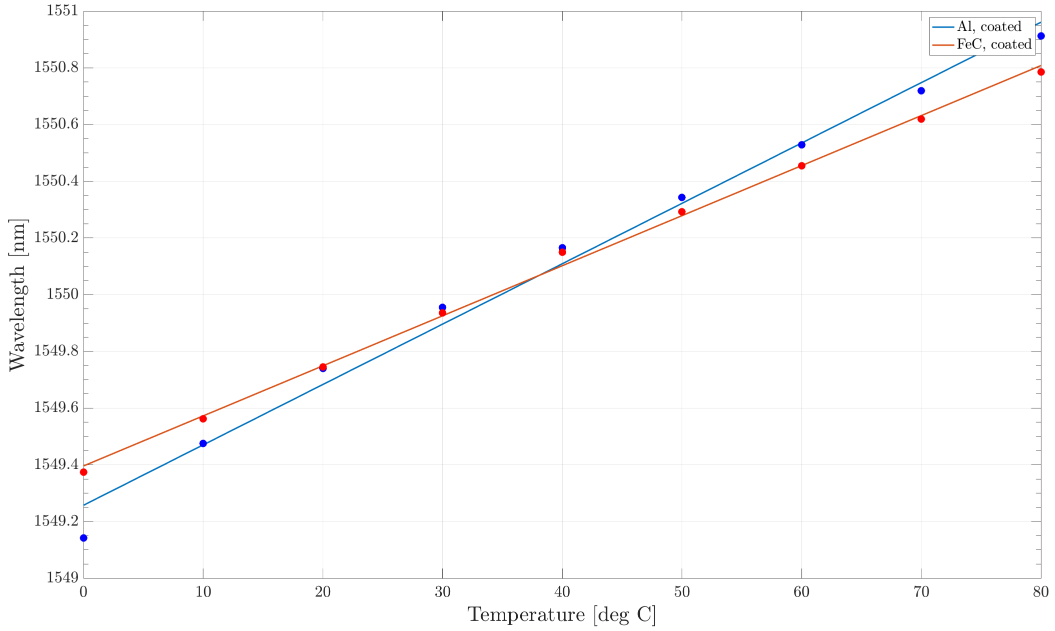


Fig. 2.36 Linear correlation for coated sensors

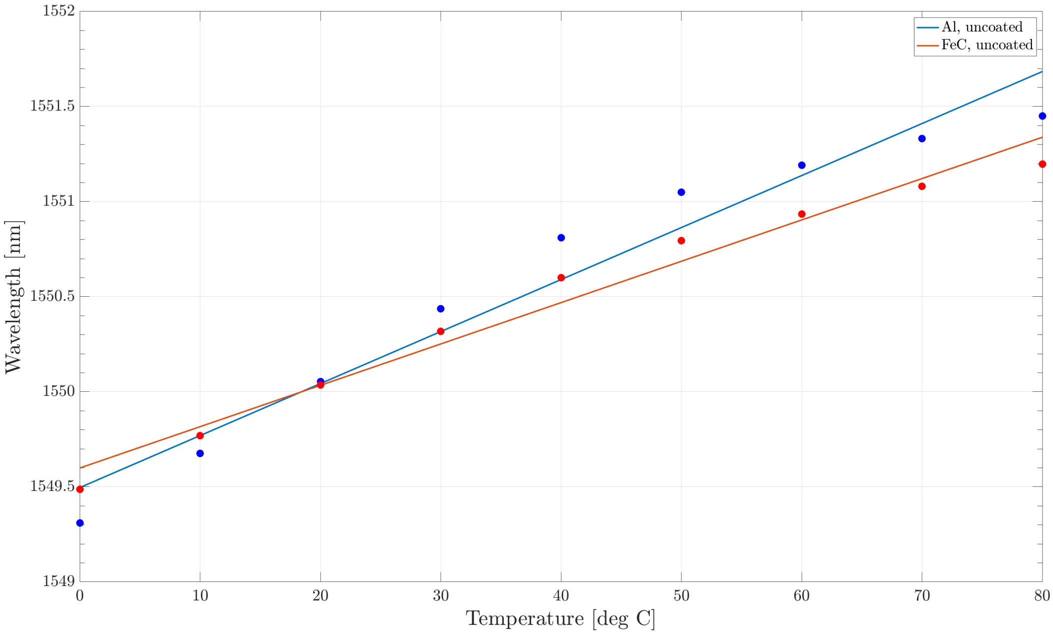


Fig. 2.37 Linear correlation for uncoated sensors

For the reasons explained before, the linear fit is accurate for the coated sensor, having coefficient of determinations $R^2 = 0.9979$ and $R^2 = 0.9902$ for steel and aluminum respectively, while for uncoated sensors is not accurate enough, with coefficient of determinations values of $R^2 = 0.9755$ and $R^2 = 0.9563$, again for steel and aluminum respectively .

In any case, the bonding coefficient as defined previously is numerically evaluated by inverting Equation 2.31, thus obtaining:

$$k_B = \frac{\frac{\Delta\lambda}{\lambda} - k_{T,fiber}\Delta T}{k_\varepsilon \alpha_{al} \Delta T} \tag{2.35}$$

using the following parameters: $k_{T,fiber} = 9.765 \cdot 10^{-6}$ °C^{-1}, $k_\varepsilon = 0.784$ obtained using equations and parameters values reported in [46].

Thus, applying the previous equation the graph shown in Figure 2.38 can be obtained. As expected, the bonding coefficient is not constant with regards to the temperature. Furthermore, an anomaly is visible close to the reference temperature, which is $T_0 = 33.9$ °C for the coated version and $T_0 = 18.9$ °C for the uncoated one. Reference temperature is defined as the temperature where the wavelength is equal to the nominal one, which for all sensors tested in this experiment is 1550 nm.

For the uncoated sensors, the values descends quite rapidly after 40 °C, and this phenomenon can be explained by referring to Fig. 2.37: after this temperature FBG readings start to plateau and linearity is lost - which is directly reflected in the bonding coefficient.

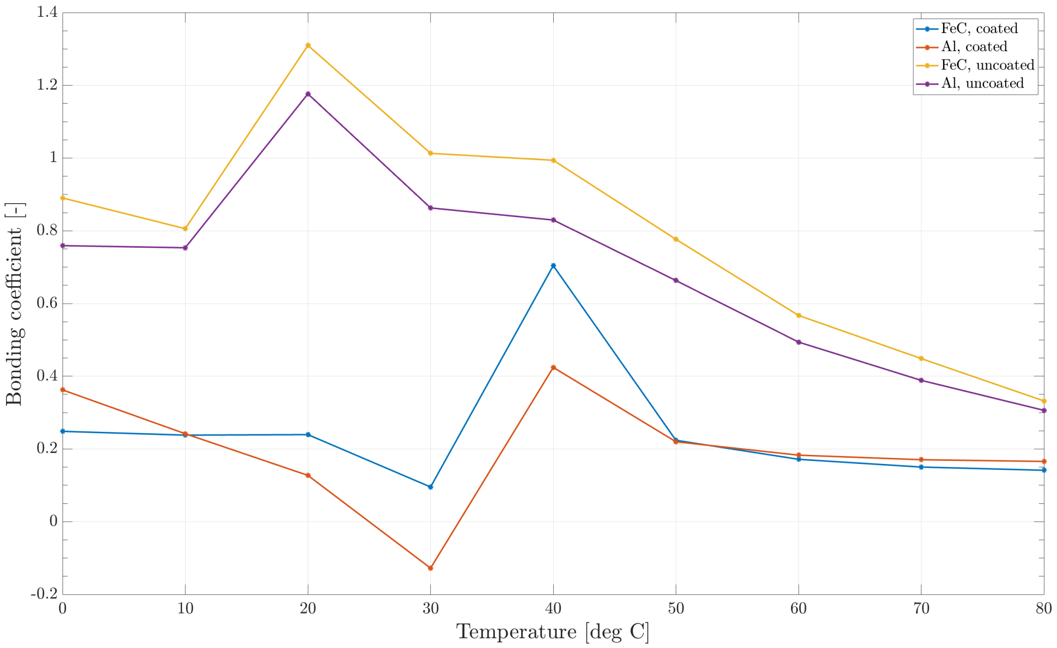


Fig. 2.38 Bonding coefficients plots

Table 2.6 Bonding coefficients

Temperature [°C]	Al, coated	FeC, coated	Al, uncoated	FeC, uncoated
0	0.3624	0.2485	0.7592	0.8903
10	0.2419	0.2381	0.7534	0.8059
20	0.1273	0.2395	1.1766	1.3099
30	-0.1278	0.0954	0.8629	1.0131
40	0.4244	0.7043	0.8297	0.9940
50	0.2199	0.2239	0.6633	0.7767
60	0.1830	0.1716	0.4939	0.5673
70	0.1705	0.1503	0.3887	0.4486
80	0.1657	0.1414	0.3063	0.3322

The most promising result is achieved by the acrylate-coated FBG bonded on a steel support - blue line in Fig. 2.38. Besides the anomaly in proximity of the reference temperature, the behavior is quite linear with an average value of $k_B = 0.246$. All the values are shown in Table 2.6.

From this preliminary results, it is clear that further investigation of the interaction between fiber, bonding resin and support is necessary. With the results evaluated in this study, the application of the formulation proposed in the previous section is not direct, as the bonding coefficient is not constant but is rather a function of temperature. Thus, a modification of the proposed model is necessary to include a temperature dependency on the bonding coefficient, while at the same time the effect of resin will have to be investigated to mitigate non-linearities, if possible, especially around the normalization temperature, which could also be an artifact of the model itself.

Chapter 3

EMA numerical models

Developing prognostics and diagnostics strategies requires a large amount of data, which is not an easy task when the system of interest is complex. Furthermore, the most useful data obtainable from a real system are run-to-failures, where the component or system is continuously operated in conditions mirroring operating conditions, with or without accelerated degradation.

As expected, run-to-failures require a number of different system to be destroyed, which is expensive and time consuming. A tool that can ease the situation is the adoption of numerical models representing the system in different health statuses and operating conditions.

3.1 Usefulness of numerical models

As previously stated, numerical models can be helpful tools in the context of prognostics and diagnostics. Numerical modeling has a long history in engineering.

Starting with the development of computers in the late 1940's, solving numerical models has become feasible when no analytical solution exist, which is generally the case for most non linear systems. In fact, the development of the first general purpose computer in the modern sense, ENIAC, was driven by the United States army need to precisely calculate artillery tables [47].

Computer simulations have been instrumental to the space race. In fact, no closed solutions exist to the gravitation equations when considering more than two bodies.

The only way to estimate the future condition of an object in space when subject to gravity from more than two bodies, is to numerically propagate the initial conditions using a simulation, which is generally referred to as a numerical propagator. Without computers, it would have been much harder to reach the Moon or plan missions outside the Earth sphere of influence. Especially for long duration missions, a high accuracy is paramount, as a small initial error can be propagated and amplified more and more as time increases, providing solutions can be completely different from reality.

Nowadays, with computers being several order of magnitudes more capable than the original ENIAC, it is possible to model, with a reasonable accuracy, planet-scale phenomena such as weather, climate and pollution spread.

Regarding this work, numerical modeling has been used to model an aeronautical electromechanical actuator (EMA), actuated by a Permanent Magnet Synchronous Motor (PMSM), in order to generate data at different levels of damage to create a system degradation dataset, as will be explained in the next chapter. In this chapter, starting from a reference HF model, validated on a real test bench, several reduced order model with different levels of fidelity will be presented.

3.1.1 Reduced order models of EMA

As previously explained, numerical models are very useful to have a digital surrogate of a physical system. However, since every model is an approximation of a real system, different levels of fidelity can be achieved, depending on the simplifying hypotheses applied during model construction.

In this context, reduced order model are useful for different reasons. The main advantage is a lower computational load on the system, since detailed modeling of some components is omitted because is deemed as not necessary for the application, as will be described shortly.

For the following models, the common approximation is considering elements with lumped parameters, i.e. intrinsic properties such as mass, resistance etc. is ’concentrated’ in a single element rather than be distributed as in a real system. It is a very common approach as it strongly simplifies the modeling of the system and is the basis to Simulink-style, block-driven modeling. On the other hand, using distributed parameters would lead to more complex and specialized techniques such as Finite

Elements Method (FEM) or Computational Fluid Dynamics (CFD) which are much harder to correctly set-up and extremely hardware-intensive.

3.2 Reference High Fidelity (HF) PMSM model

The High Fidelity model will be used as a reference to test the simplified models against and evaluate accuracy, as the HF model has been optimized on a physical test bench to match the results as closely as possible. The test bench is shown in Fig. 3.1 (from [48]). Calibration details can be also found in [48].

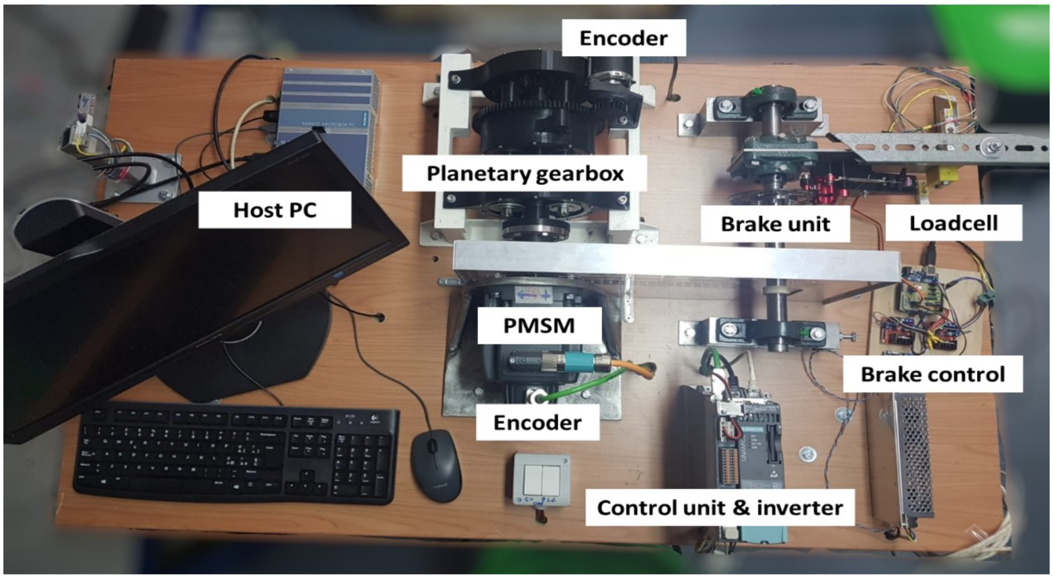


Fig. 3.1 Test bench for EMA model

As previously described, the model represents an electromechanical servoactuator driven by a Permanent Magnet Synchronous Motor (PMSM), and includes non-linear phenomena such as dry friction, noise and backlash. The main approximation is, as discussed before, lumped parameters.

Additionally, the rotor-stator electromagnetic interaction is not explicitly modeled but the following equation is used:

$$T_m = \Sigma k_j i_j \tag{3.1}$$

where k_j and i_j are the phase motor constant and phase current, respectively. This approximation is needed to avoid using electromagentic FEM tools to resolve the motor-stator coupling that would render the model extremely complex and slow to simulate.

In Fig. 3.2 the top level view of the model is shown. The **Com** block is the programmable command block used to create the command imposed on the motor, but referred to the slow shaft, as the servomechanism is a position-commanded one.

The **Ext. load** is a programmable external load block used to model a torque acting externally on the servo, which is representative of phenomena like hinge moment as this servomechanism is referred to an aerospace one.

The **HF model** is the core of the model and will be described in the following subsections, as it is complex and is composed by several nested subsystems. Finally, the scope on the right is used to visualize important parameters such as motor position (*theta_m*), motor speed (*Dtheta_m*), user position (*theta_u*) and equivalent single-phase current (*I_3eq_HF*) which is used for comparison with the other single-phase equivalent models.

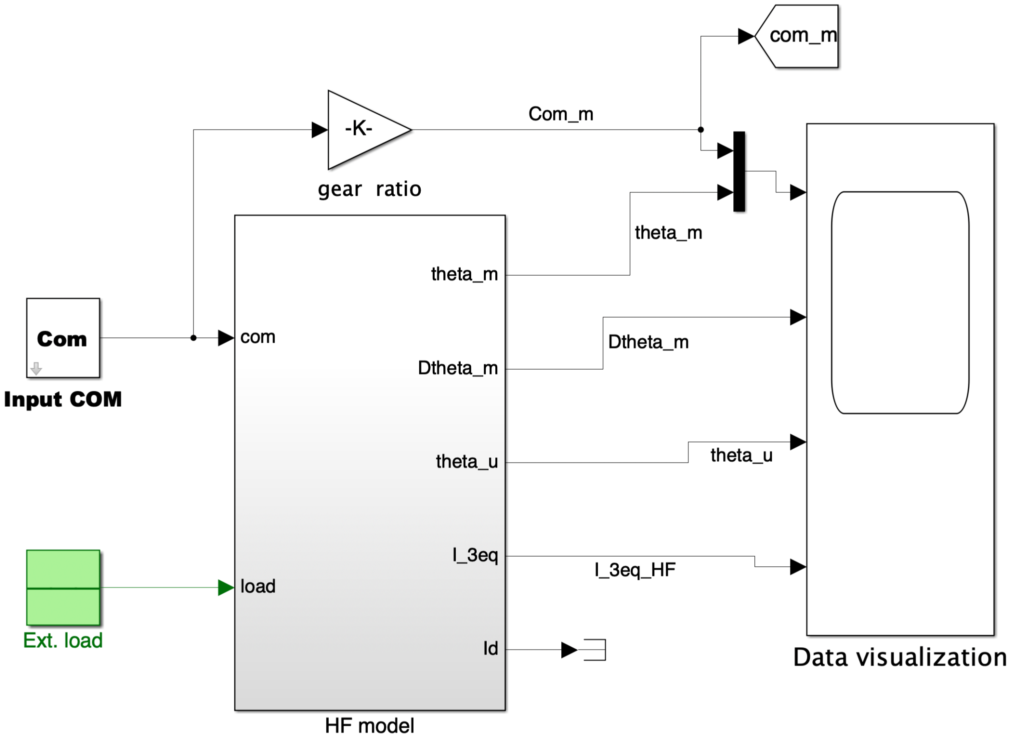


Fig. 3.2 HF model top level view

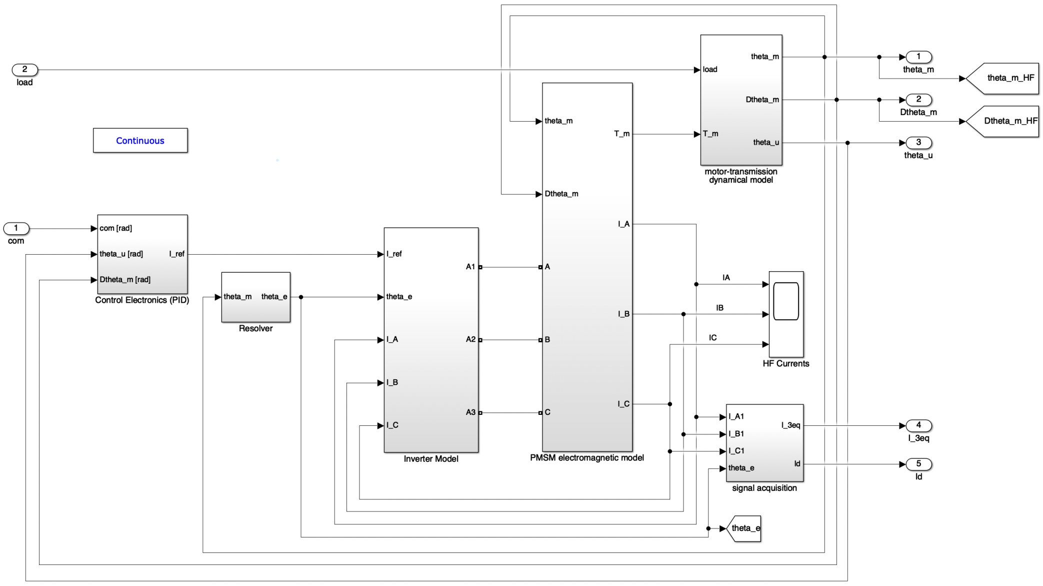


Fig. 3.3 HF model overview

In Fig. 3.2 the first level of the HF model subsystem is shown. Subsystems will be briefly described here, as the expanded view and a more in-depth description will be provided for each of them. From left to right:

- **Command Electronics (PID)**: this subsystems contains the control logic for the whole servomechanisms. The main control element, as the name implies, is a PID (Proportional-Integral-Derivative) controller, with several added functions such as anti-windback. Additionally, there are saturations for several quantities such as driving current and motor velocity.
- **Resolver**: this simple subsystem converts the mechanical angular position of the rotor to electrical position considering the number of pole pairs.
- **Inverter model**: this subsystems models to the component level the inverter used to drive the PMSM motor. It receives a command current, *I_ref*, applies the Clarke-Park transformation, to create the three current signals *A1, A2, A3* which are created using Simscape Power Systems, in order to create a physical representation of the electrical part. The three measured phase currents *I_A, I_B, I_C* are also inputs to the system as they are used as feedback.
- **PMSM electromagentic model**: in this subsystems, the motor-stator interaction is simulated. Additional inputs besides the driving signals are motor

position and speed (*theta_m, DTheta_m*). The main output of the subsystem is the motor torque T_m, which is fed through the next subsystem representing the transmission model. Finally, the currents are also passed forward for logging and plotting.

- **Motor-transmission dynamical model**: it models the geared transmission from motor shaft to user shaft, while including phenomena such as viscous and dry friction, and backlash.
- **Signal acquisition**: this systems is used to filter and log the current signals.

3.2.1 Command electronics (PID) subsystem

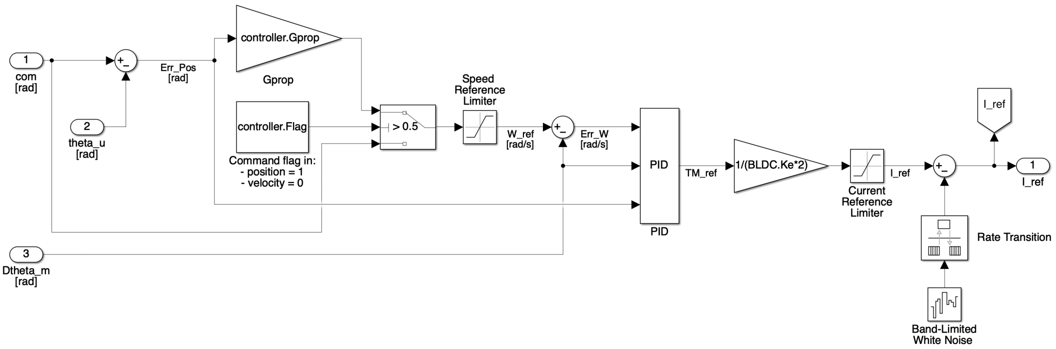


Fig. 3.4 Command electronics (PID) subsystem

The structure of the subsystem is shown in Fig. 3.4. The main input to the controller subsystem is the user command position setpoint, which is input 1 on the left of the figure. The error is promptly calculated using the measured user position, which is input 2.

The error is now multiplied by the controller proportional gain, *controller.Gprop*. The flag *controller.Flag* is used to switch the controller behavior from position control to speed control. For the purposes of this thesis, only position control has been used.

The position error, now multiplied by the controller proportional gain (with units s^{-1}), is the reference speed *W_ref*, which is saturated by the *Speed Reference Limiter* saturation block.

The internal speed feedback loop is now closed and the resulting speed error is calculated (*Err_W*). The speed error, position error *Err_Pos* and measured speed *DTheta_m* (input 3) are fed to the PID subsystem, which is a saturated PID controller with several anti-windup solutions implemented.

The output of the PID subsystem is the reference torque *TM_ref*, which is multiplied by the factor $(2 \cdot \mathrm{BLDC.Ke})^{-1}$ to obtain the reference current to drive the motor. It stems directly from the motor equation:

$$T = K_T \cdot i = K_e \cdot i \tag{3.2}$$

where T is motor torque, i is current, K_T is the motor torque coefficient and K_e is the back-EMF coefficient. It is assumed that $K_T = K_e$, and thus the resulting equation. From the data available, *BLDC.Ke* represents the back-EMF constant of a single branch, thus the multiplication by 2 in the previous equation, since $BLDC.Ke = K_e/2$.

The torque calculated is now passed through a saturation block, Current Reference Limiter, to implement the overcurrent protection for the motor, and thus obtaining the saturated reference control current *I_ref*.

Finally, band-limited white noise is added to the reference current to simulate noise along the whole controller subsystem, with a rate transition to interface it properly as the white noise can have a different sample time with respect to the other blocks running at the model timestep. In this case, the noise sample rate is set to 100 model timesteps.

PID block

The original PID (green area in Figure 3.5) was developed, along with the model, before the test bench was available, and is thus implemented in the parallel PID form, altough using the characteristic times formulation rather than the more traditional P, I, D coefficient formulation.

Starting with the (user) speed error, the proportional gain *controller.PID.GAP* $[\mathrm{N\,m\,s/rad}]$ is applied, obtaining the proportional component of the command.

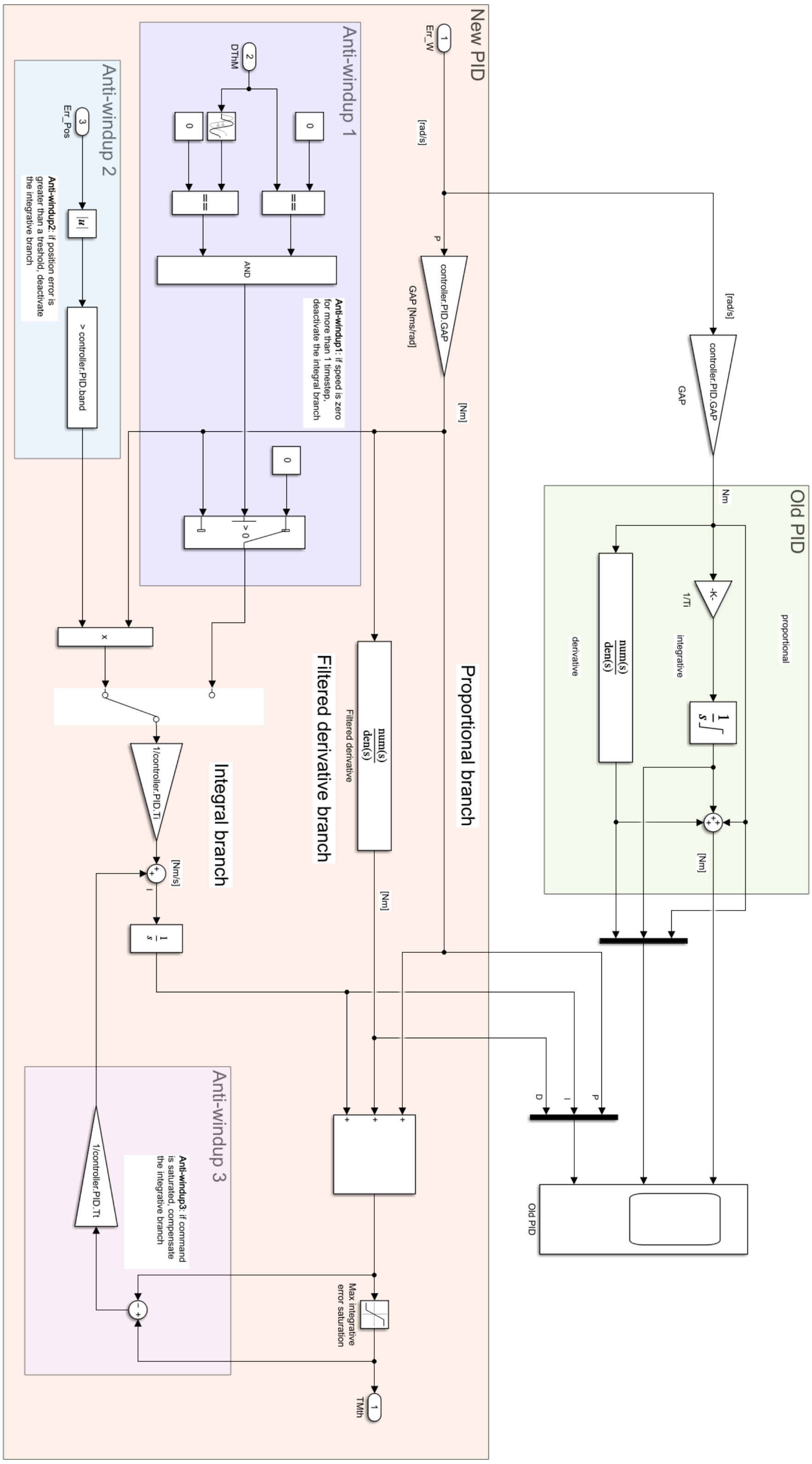


Fig. 3.5 PID subsystem

The same value is divided by *controller.PID.Ti*, where *Ti* is the integral time constant of the controller. The output is then integrated and saturated to obtain the integral component of the controller output.

Finally, a transfer function is used to model the filtered derivative part of the PID, which is in the following form:

$$H(s) = \frac{controller.PID.Td \cdot s}{\frac{controller.PID.Td}{controller.PID.N} \cdot s + 1} \tag{3.3}$$

where *controller.PID.Td* [s] is the derivative characteristic time and *controller.PID.N* [s^{-1}] is the derivative bandwidth.

The three components are then summed to obtain the total output of the PID controller.

However, the PID as formulated does not have any anti-windup implemented, but only an output saturation. During experimental validation, it has been found that the lack of an anti-windup caused discrepancies between the model and the test bench.

Thus, three different anti-windup have been implemented, using different triggering logic, in the New PID architecture (red area). Starting with the Anti-windup 1 (violet area), the triggering logic is that if the speed is zero for more than a single timestep, the integrative branch should be deactivated. This case can occur when the servomechanism is at one edge of the actuation range and further rotation is not possible, thus a dangerous compounding of the integral error is avoided.

Anti-windup 2 has simpler logic, i.e. the integrative branch is deactivated when the position error is greater than a treshold. Again, this is done in order to avoid total controller saturation caused by excessive integrative error compounding.

Finally, anti-windup 3 (pink area) checks whether the saturated command is equal to the non-saturated one, i.e. it checks if the controller is in saturation conditions. If this is the case, then the integrative branch is deactivated as additional summing will lead to windup.

It has to be noted that in this model system 1 and 2 do not operate at the same time, and only one of either is active at the same time. In this thesis, only system 2 was selected. On the other hand, system 3 is always active independent of which system between 1 and 2 is selected.

3.2.2 Resolver

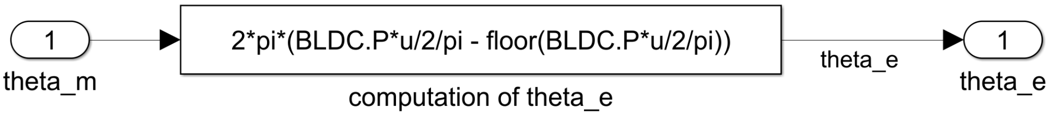


Fig. 3.6 Resolver subsystem

The resolver subsystem (Fig. 3.6) is simple, as it only computes the electrical angle using the mechanical angle of the motor and the number of pole pairs, using the following equation:

$$\theta_e = 2\pi\left(\frac{P\cdot\theta_m}{2\pi} - \text{floor}\left(\frac{P\cdot\theta_m}{2\pi}\right)\right) \tag{3.4}$$

where *BLDC.P* = *P* is the number of pole pairs of the motor, *theta_e* is the electical angle and *theta_m* is the mechanical angle. In other words, this function wraps the mechanical angle between 0 and 2π after scaling by the number of pole pairs, as the electrical angle is the mechanical angle multiplied by the number of pole pairs, as more polar expansion means less mechanical rotation is needed before each commutation.

3.2.3 Inverter model

The inverter subsystem models the electronic driver (an H-bridge) including the commutation logic used create the three driving currents starting from the reference current originating from the controller subsystem. As shown in Figure 3.7, several subsystems are present inside, performing different functions.

The motor is controlled using Field Oriented Control, a technique that allows easier management of the phase switching as it performs a coordinate transformation between the *a, b, c* rotating frame of the three phases to a non-rotating *q, d, 0* frame aligned with the magnetic axis of the rotor. Using two different transformations, the rotating *a, b, c* frame of reference is firstly transformed in the instantaneous direct and quadrature flux axis using the Clarke transform obtaining the $\alpha, \beta, 0$ frame of

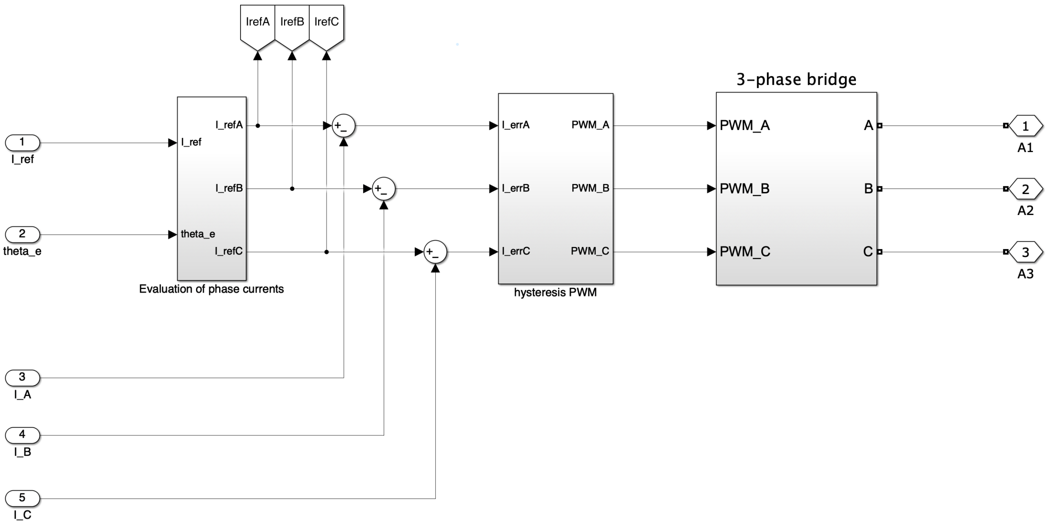


Fig. 3.7 Inverter subsystem

reference. Then, by application of the Park transformation, the frame of reference is made stationary along the direct and quadrature axis of the motor by rotating it of θ_e which is the motor electrical angle.

In mathematical terms, the two change of coordinates can be expressed as:

$$[d,q,0] = K_P \cdot [\alpha,\beta,0] = K_P \cdot K_C \cdot [a,b,c]$$

The two transformation matrices are the following:

$$K_C = \sqrt{\frac{2}{3}} \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \qquad K_P = \begin{bmatrix} \cos(\theta_e) & \sin(\theta_e) & 0 \\ -\sin(\theta_e) & \cos(\theta_e) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.5}$$

However, on the control side, the inverse transformations are used (Fig. 3.9), since the commanding variable is *I_ref*. In general, both I_q and I_d can be controlled, being the torque component and the flux component. In this case, the flux component is not of interest since no field weakening is applied and thus *I_ref* is set equal to I_q, thus all the reference current contributes to an increase of the motor torque output. The equations implemented in the model are simplified version of the general form that can be obtained from inverting Eqns. 3.5, since applying the inverse Park transformation gives a component which is always equal to zero.

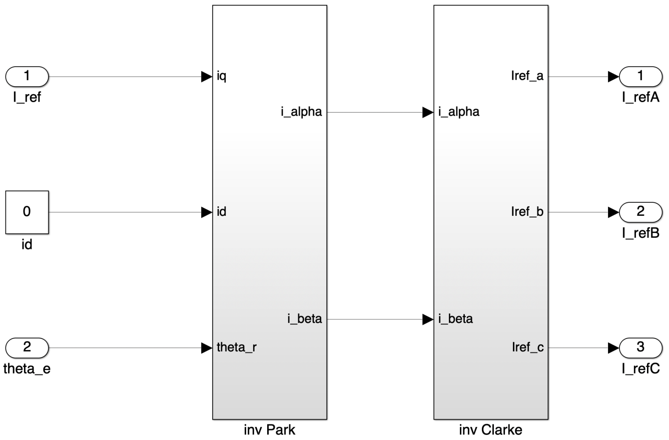
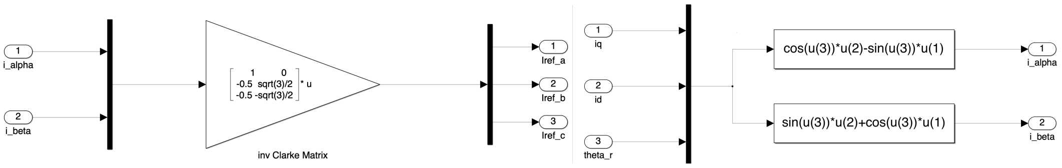


Fig. 3.8 Phase evaluation subsystem

The direct transformations are instead used on the feedback loop to read the actual motor currents values and used them for error calculation purposes, going from the $[a,b,c]$ to the $[d,q,0]$ frame of reference.

In the model, the *Evaluation of phase currents* subsystem (Fig. 3.8) uses the inverse Clarke and Parke transformations to obtain the reference currents referred to the three motor phases. To do so, the *I_ref* coming from the controller is considered as the quadrature current using the inverse Clarke transformation (Fig. 3.9a) as previously explained. After both transformations, the three reference currents in the $[a,b,c]$ frame of reference are obtained and will be used to drive the motor.



(a) Inverse Clarke transformation

(b) Inverse Park transformation

Fig. 3.9 Inverse Park/Clarke transformations

Once the three reference currents have been obtained, the three individual current errors are evaluated by subtracting measured value from reference. Then, the error is used to drive an hysteresis PWM, as in Fig. 3.10.

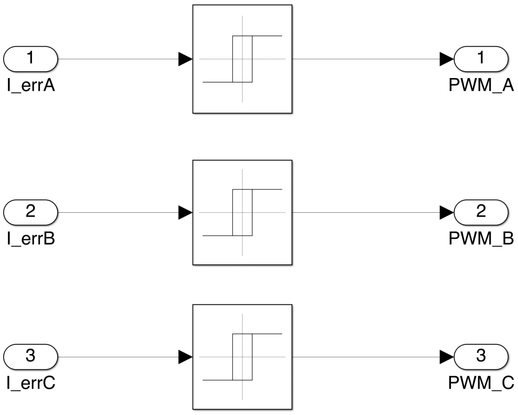


Fig. 3.10 Hysteresis PWM

The pulse generation is a simplified version of the common carrier-wave PWM with fixed frequency. In this case, there isn't a pre-defined PWM frequency, as the signal is triggered whenever the error is above (or below) the current dead-zone set up in the block. Modeling the oscillating trigger in this way as the advantage of being very easy to implement, but on the other hand, the lack of a reference carrier frequency makes the triggering as short-timed as the model timestep. Additionally, for very tight dead-zone width, the model will become extremely stiff since there will be a switching between high and low values almost every timestep.

This slows down the simulation and will need to be evaluated in the future, as in general a dedicated PWM subsystem should improve simulation time.

Finally, the three switching signals are used in the *3-phase bridge* subsystem to drive an electrical model of the 3-phase H-bridge, as in Figure 3.11.

The upper portion simply create a negation of the three command signals using *NOT* blocks, while the original signal is casted as boolean. The six signals are then grouped with a bus and used as inputs for the MOSFET-based Simscape Electrical Power Systems *Universal Bridge*, which is connected to a DC Voltage source.

The three output signals are shown on the right and are not simple logical, one-directional signals, but are rather physical connection capable of bi-directional energy transfers. These values will be directly connected to the phases coils in the *PMSM electromagnetic model* subsystem that will now be described.

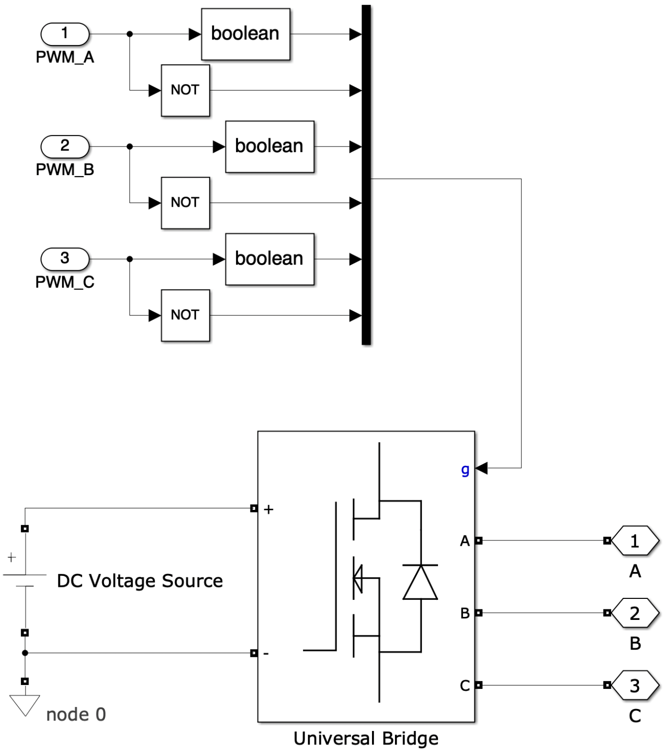


Fig. 3.11 3-phase bridge

3.2.4 PMSM electromagnetic model

This subsystems is used to model the electromagnetic interaction between the stator and rotor of the PMSM motor, in a simplified way. As shown in Figure 3.12, several subsystems are present, starting with the *computation of back-EMF coefficients*.

Since this motor is a PMSM rather than a more common Brushless DC (BLDC), the back-EMF profile is not trapezoidal as function of the angle, but is rather sinusoidal. Thus, using the mechanical angle to calculate the electrical angle, a simple modulation is used to create three 120° out-of-phase sinusoids representing the three phases back-EMF coefficients (Fig. 3.13).

Computation of back-EMF coefficients

Additionally, since this model is intended to simulate also faulted conditions, an additional modulations is introduced for each of the three coefficients as function of the static eccentricity parameterized using ζ and ϕ. Eccentricity magnitude ζ is

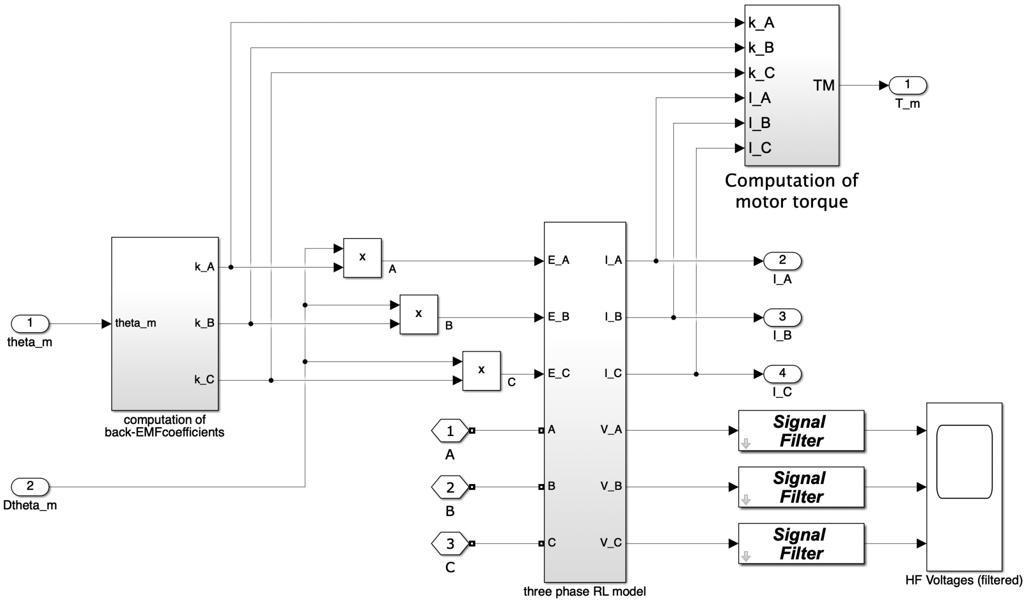


Fig. 3.12 PMSM electromagnetic model subsystem

defined as $\zeta = x_0/g_0$, where x_0 is the rotor axis linear offset from intended position and g_0 is the nominal air gap, i.e. when eccentricity is zero ($x_0 = 0$) . Eccentricity phase ϕ is the the angle where the air gap is minimized, using an horizontal axis as zero and counting in anti-clockwise direction. Full derivation can be found in [48].

The three f(u) blocks that modulates the back-EMF coefficients are the following:

$$\text{(A) - } \; BLDC.Nabc(1) * BLDC.Ke * u(2) *$$
$$(1 + (BLDC.P > 1) * BLDC.zeta * cos(u(1) - BLDC.phi))$$
$$\text{(B) - } \; BLDC.Nabc(2) * BLDC.Ke * u(2) *$$
$$* (1 + (BLDC.P > 1) * BLDC.zeta * cos(u(1) - BLDC.phi + 2 * pi/3))$$
$$\text{(C) - } \; BLDC.Nabc(3) * BLDC.Ke * u(2) *$$
$$* (1 + (BLDC.P > 1) * BLDC.zeta * cos(u(1) - BLDC.phi + 4 * pi/3))$$

where *BLDC.Nabc* is a three-element vector with the fraction of active turns per each coil, *BLDC.Ke* is the nominal back-EMF coefficient, *BLDC.P* is the number of pole pairs, *BLDC.zeta* is the non-dimensional eccentricity as explained above and *BLDC.phi* is the eccentricity phase. *u(1)* and *u(2)* are the relative inputs to the function block.

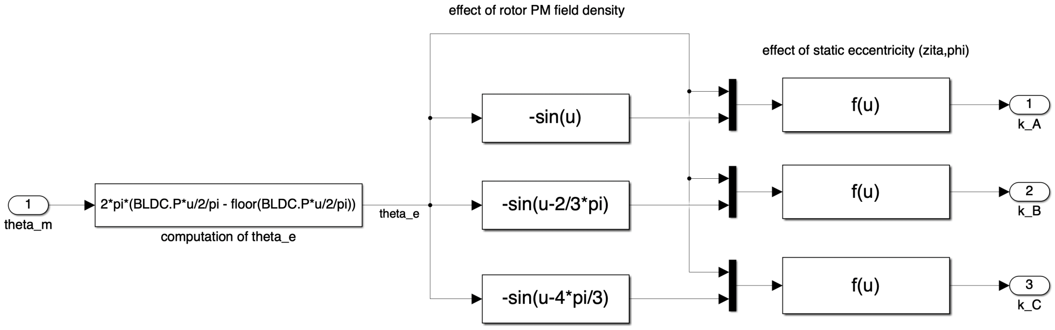

effect of rotor PM field density
effect of static eccentricity (zita,phi)
1
theta_m
2*pi*(BLDC.P*u/2/pi - floor(BLDC.P*u/2/pi))
computation of theta_e
theta_e
-sin(u)
-sin(u-2/3*pi)
-sin(u-4*pi/3)
f(u)
f(u)
f(u)
1
k_A
2
k_B
3
k_C

Fig. 3.13 Back-EMF coefficient evaluation

Once the back-EMF coefficients have been evaluated, they are multiplied by the motor angular speed to obtain the back-EMF voltages to be applied to the stator coils.

Three phase RL model

This subsystem is relatively simple and is modeled by using three RL branches (Fig. 3.14) representing the three phases, as it is usually done, wound in a wye configuration. Additionally, the motor center is not grounded as typical for aerospace applications.

Additionally, one controlled voltage source per branch is added with opposite polarity to the main supply; this represents the back-EMF component. The value has been calculated in the previous subsystem and is updated every timestep.

Voltage and current monitors are present to measure these variables of interest.

Motor-transmission dynamical model

The transmission model is relatively complex as it includes several phenomena such as viscous friction, backlash, and an improved dry friction model, the Borello model [49].

Starting from the left hand side of the model, the external load is reduced to the fast shaft by multiplying by the gear ratio *dyanmics.tau*. The first sum block

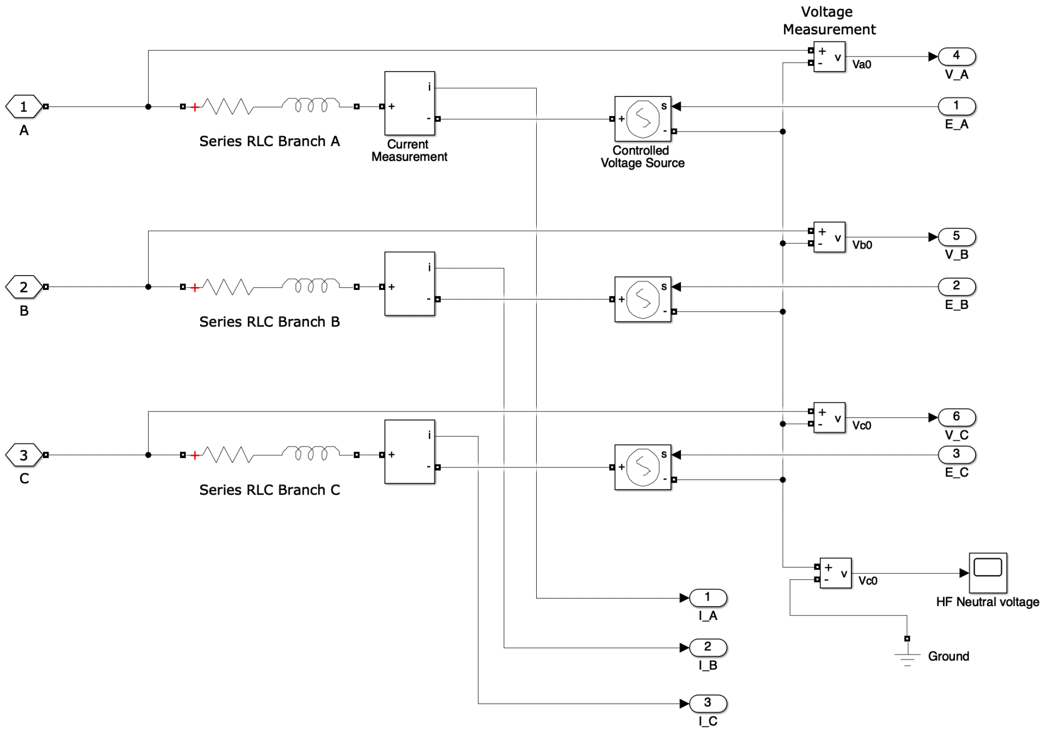


Fig. 3.14 Three-phase RL model

evaluates the torque balance as:

$$T_e = T_m - L - f_v \tag{3.6}$$

where T_e is the effective torque acting on the transmission, T_m is the motor torque and f_v is the sum of the viscous friction components, one of directly proportional to motor speed, with coefficient *dynamiucs.CU* and one to the signed square root of speed, with coefficient *dynamics.CM*. The second term has been empirically added during experimental validation.

The second sum block subtracts the dry friction calculated using the Borello friction model (Fig. 3.16), which is an implementation of the Coulomb model intended for numerical simulation, as it provides a robust behavior even in proximity to the static-dynamic boundary, where purely analytical models fail. In short, this dry friction model evaluates the acting (or effective) torque on the transmission, and if that is lower than the maximum static friction force, it sets it equal to the acting torque. Once the acting torque is greater than the maximum static friction, the system starts sliding and a new value of dynamic friction is calculated; *reducer*

Stribeck and *brake Stribeck* are two relatively small contribution that appears in conditions of incipient motion, and are also implemented in the model to have a more realistic representation of the friction force change in proximity of the stick-slide boundary. In fact, a discontinuity in the friction force is not physical and the Stribeck contributions create a continous and more realistic friction curve.

Additionally, a zero-crossings check is made on the motor speed; if a zero crossing is detected, the speed signal is reset to zero, as it implies that the conditions changed to sticking rather than sliding at least for a timestep.

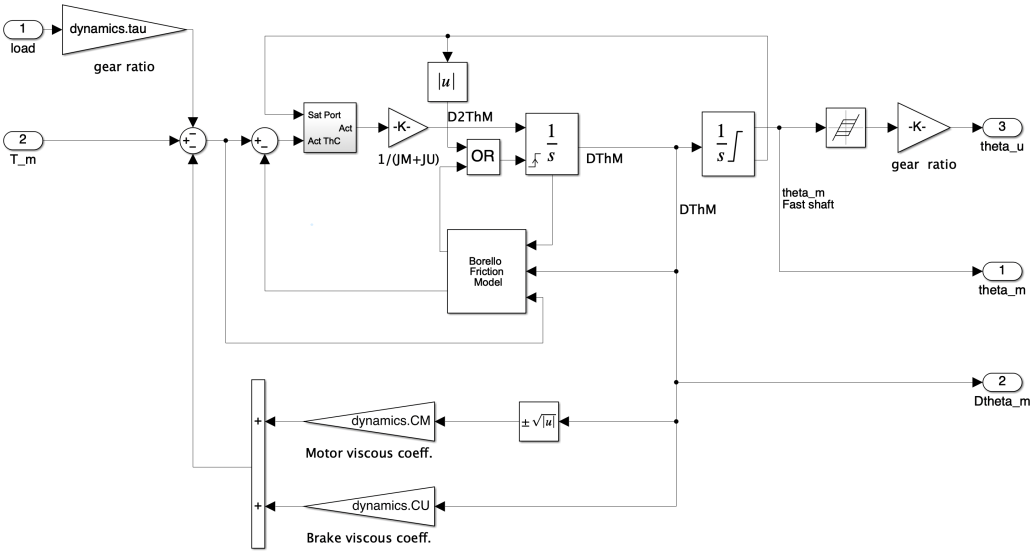


Fig. 3.15 Motor-transmission dynamical model

After that, a saturation port is used to stop the integration when the end stops are reached, so neither position or speed can increase.

After scaling using the sum of motor and user inertia (reduced to fast shaft, i.e. dividing by *dynamics.tau*^{2}), the first integrator uses this acceleration value to calculate the motor speed, *DThM*. The *OR* block before provides a speed reset signal when either the position integrator provides a saturated value, i.e. an end stop has been reached, or when the Borello friction model provides a signal. This happens when the system starts moving so the rising edge state port on the speed integrator triggers.

The speed signal can now be integrated to obtain an angular position, and this is done using the second cascaded integrator. The position integrator is saturated between the two end stops. Finally, backlash can be applied (but in this thesis has not been investigated, i.e. has been set to zero) before obtaining the user position dividing by the gear ratio.

On the lower side, as previously stated, the two components of the viscous friction have been evaluated by multiplication of motor speed with *dynamics.CU* for the true viscous component and by *dynamics.CM* after signed square rooting for the quasi-viscous component.

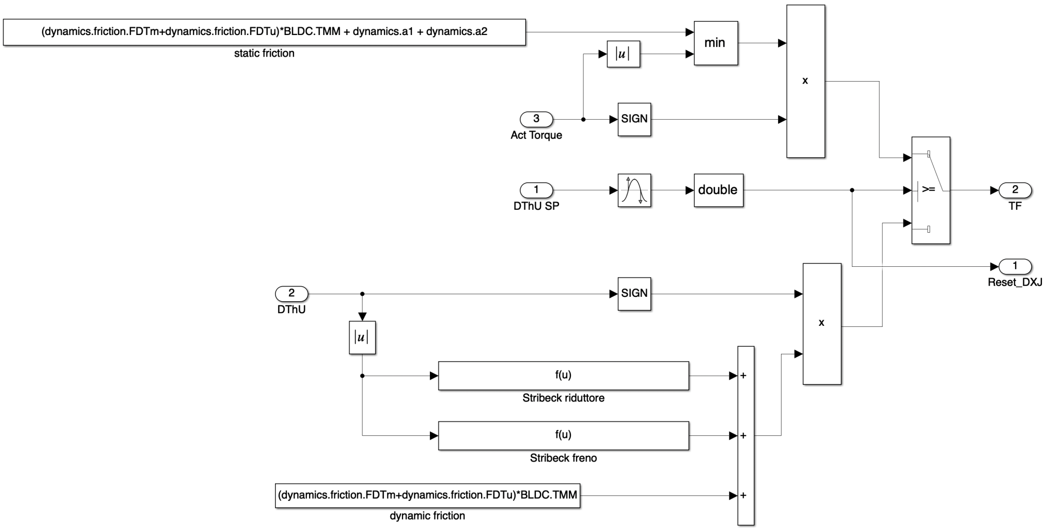

(dynamics.friction.FDTm+dynamics.friction.FDTu)*BLDC.TMM + dynamics.a1 + dynamics.a2
static friction
|u|
min
x
3
Act Torque
SIGN
1
DThU SP
double
>=
2
TF
1
Reset_DXJ
2
DThU
SIGN
x
|u|
f(u)
Stribeck riduttore
f(u)
Stribeck freno
(dynamics.friction.FDTm+dynamics.friction.FDTu)*BLDC.TMM
dynamic friction
+
+
+

Fig. 3.16 Borello friction model

Computation of motor torque

This is the simplest subsystem which evaluated the torque generated by the motor by knowing the currents in each phase. It simply applies the equation:

$$T_m = k_A \cdot i_A + k_B \cdot i_B + k_C \cdot i_C \tag{3.7}$$

where k_j is the instantaneous back-EMF coefficient for the phase and i_j is the phase current. To be more accurate, the coefficient considered should be the motor torque coefficient, usually K_τ (with units Nm/A), which is in fact defined as the ratio between motor generated torque and current applied. However, it can be demonstrated from conservation od energy that the back-EMF coefficient, usually K_V with S.I. units $\mathrm{rad}/(\mathrm{V}\cdot\mathrm{s})$, must be equal to the motor torque coefficient. This approximation is true when losses are ignored, which is a reasonable assumptions when the motor is used far from saturation conditions.

3.2.5 Signal acquisition

This simple subsystems is used to change coordinate frame to the three measured currents *I_A, I_B, I_C* to obtain *i_q* and *i_d*, i.e. the quadrature and direct current. This is achieved by successively applying the Clarke and then the direct Park transform, which are simply the inverse of those shown in Fig. 3.9. These measures are useful quantities as they can be directly compared with those from other simplified, single-phase motors that will be described in the following sections. In fact, *i_d* is also labeled as *I_3eq* as it can be seen as the single-phase current for a simplified equivalent monophase motor.

The signal filters are third-order low-pass filter obtained by cascading three first-order low-pass filter; each stage has a time constant of 50 µs.

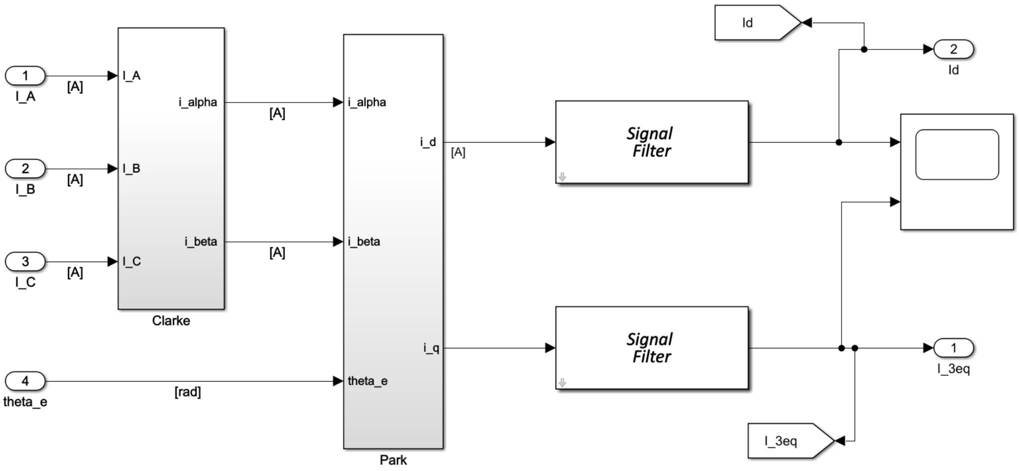


Fig. 3.17 HF signal acquisition subsystem

3.3 Medium Fidelity (MF) PMSM model

The HF model is a very precise and detailed model, and can model the behavior of the reals system with a high degree of accuracy. However, the trade-off is that the model runs very slowly even when optimized using Simulink Accelerator mode that can compile part of the model and run it without using the Java interpretation layer, thus strongly increasing performance.

Thus, the first step to improve the simulation time is to simplify the slowest subsystems of the model, which are those simulating electronic components. This is the purpose of the Medium Fidelity model, which simplifies the *PMSM electromagnetic model* and uses a finite-state machine to evaluate which phase become voltage-saturated. Subsystems that are not analyzed are to be intended equal to those found in the HF model. Additionally, the hysteresis PWM has been eliminated which is also a big contributor to computation time.

In Figure 3.18 an overview of the MF model is shown, and it can be seen the conceptual similarity to the HF model.

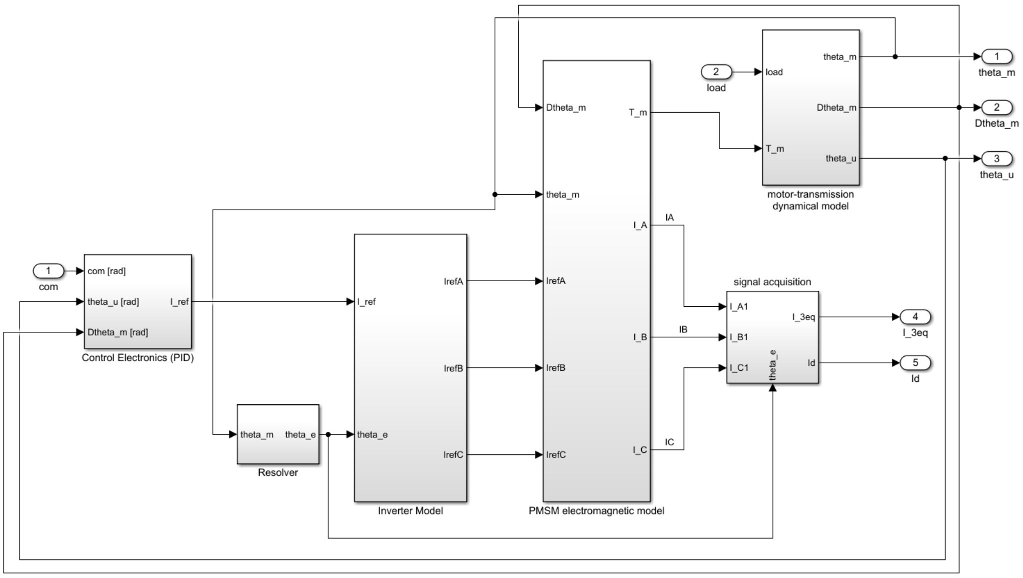


Fig. 3.18 Medium Fidelity model overview

3.3.1 MF Inverter model

As previously stated, the inverter model in the medium fidelity model (Fig. 3.19) is a simplification of the high fidelity one. The main difference is the lack of the hysteresis PWM, and also the error calculation between reference and measured currents. In the MF model, the reference currents are directly used to drive the motor computation. The difference can be attributed to how the model is set-up, as the electromagnetic part is not physical anymore but is rather evaluated using mathematical functions.

Inside the *Evaluation of phase currents* subsystem the previously explained inverse Clarke and Park transformations are applied, analogously to the HF model, to change the [d, q, 0] frame of reference the [a, b, c] frame.

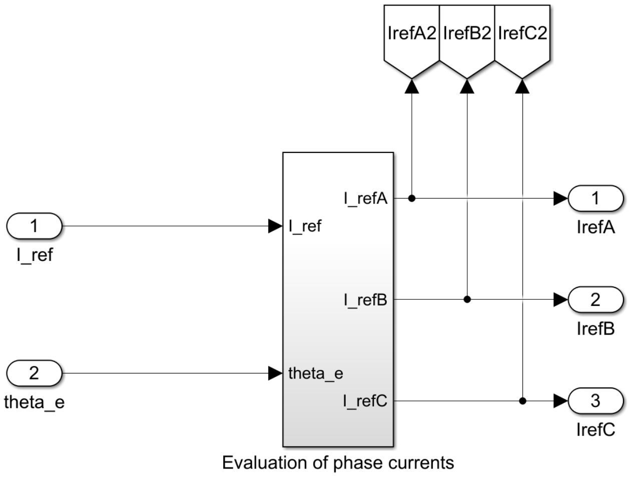


Fig. 3.19 MF inverter model

3.3.2 MF PMSM electromagnetic model

The main difference between the HF and MF model can be found in how the actual EM interaction between the stator and rotor. The physical modeling is abandoned and equations are now used to approximate the behavior of the stator phases. In Fig. 3.20 the top level view of the EM model is shown, which is now substantially different from the HF implementation.

The *computation of back-EMF coefficients* and *Computation of motor torque* are the same of the HF model and will not be explained again, since the parameters are also equal. However, the back-EMF calculation has been refactored and is now grouped inside the *Back-EMF calculation* subsystem, which simply multiplies the instantaneous back-EMF coefficients *k_A, k_B, k_C* by the motor angular speed *DTheta_m* as usual. It has been empirically found that an increase of 7.5% for the back-EMF values improves the adherence to the results provided from the HF model, so it has been applied with a gain block (with value 1.075) before passing the back-EMF values to the PMSM subsystem.

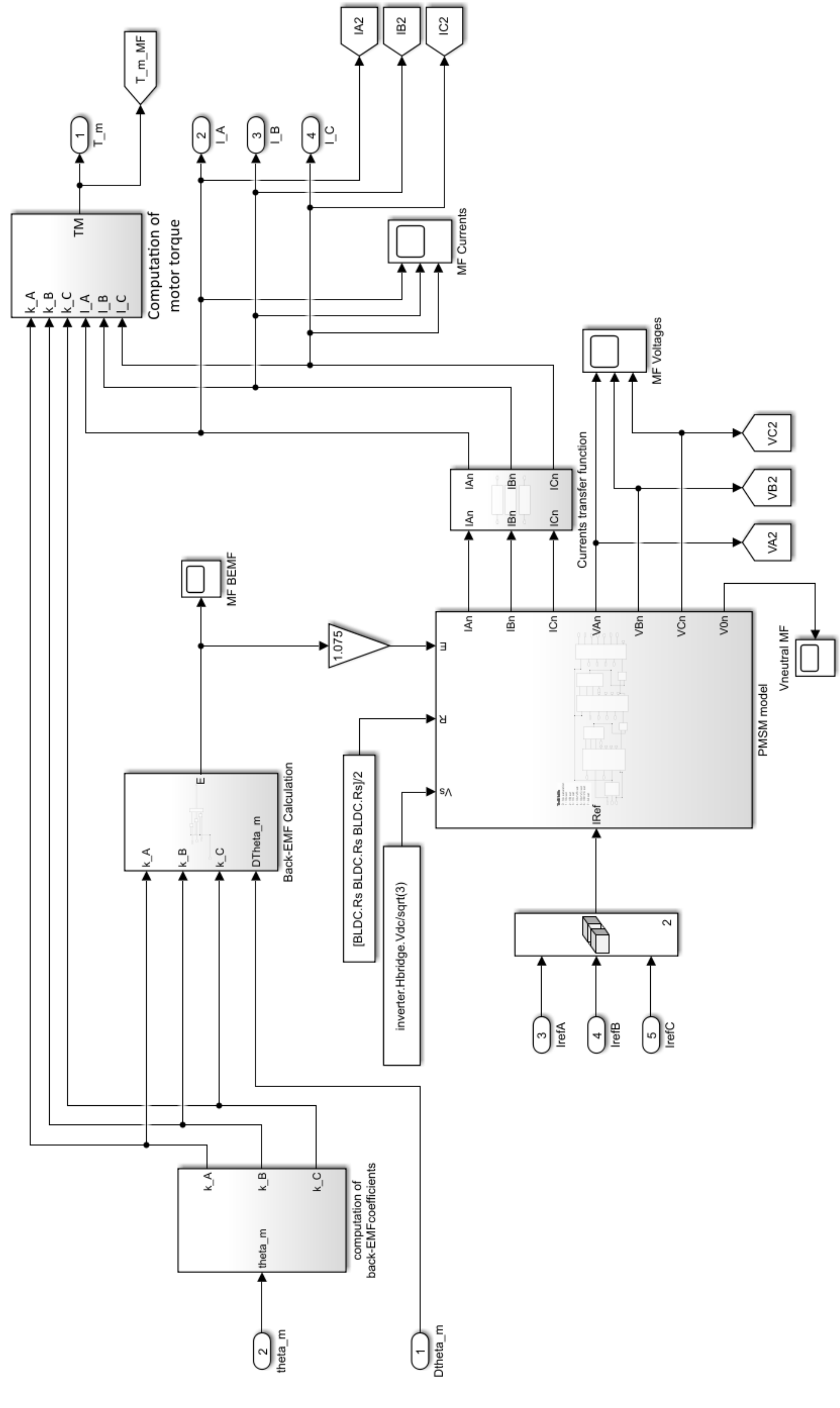


Fig. 3.20 Medium Fidelity EM model

MF PMSM model

The inputs of the PMSM model subsystem (Fig. 3.21) are the three phase resistances, which in this case are set to nominal values (*BLDC.Rs*/2), the supply voltage *inverter.Hbridge.Vdc*/$\sqrt{3}$ (i.e. 380 V phase-to-ground, which means 220 V phase-to-phase), the three reference currents *IrefA, IrefB, IrefC* obtained from the inverter subsystem, which are vectorized before being passed to the subsystem, and the three back-EMFs, which are vectorized in the vector *E* before being sent to the subsystem.

In this model, the PMSM is modeled using a finite state machine to comply with conditions of voltage saturation, since it as in important phenomena that should be taken into account for a good fidelity simulation.

Since the motor is current controlled, the three reference currents are used to calculate the three different control voltages:

$$V_j^{ref} = R_j \cdot i_j^{ref} + E_j = R_j \cdot i_j^{ref} + k_j \omega_m \tag{3.8}$$

where V_j^{ref}, i_j^{ref} are the reference voltage and current for a phase, E_j is the phase back-EMF, k_j is the back-EMF coefficient and ω_m is the motor angular speed. However, these voltages are limited by the maximum supply voltage, V_s. It is thus possible to have a condition in which one or more phases are in voltage saturation, so that phase(s) have to be set equal to $V_j = V_s \cdot \mathrm{sign}(V_{ref}^j)$.

After saturation are imposed, the electrical circuit is resolved obtaining currents and voltages for the three phases and the neutral voltage, being a wye-wound circuit. However, it is now necessary to re-evaluate the voltages obtained after the first step.

It is possible that, after one phase becomes voltage-saturated, the system imbalance could push the other phases voltage above the supply value.

The truth tables (*Case eval 1st stage*, *Case eval 2nd stage*) are used to compare the step output voltages to V_s, and a flag is used to determine which phases are in saturation.

The direct lookup tables (*2nd step case selection*, *3rd step case selection*) are then used to select which set of equations the next step will use to solve the system. These tables are used to keep track of which phases have already been set to saturated in previous steps and add the newly saturated phases. In particular, three different evaluations are needed since a condition that can occur is that only one phase is initially saturated; after system resolution, however, is found that a second phase is also in saturation; the final evaluation might show that even the third phase in saturation. Thus, only the last step currents and voltages are used as 'true' values outside of this subsystem.

For all the blocks in the subsystem, the truth table shown in Fig. 3.21 is used to maintain consistency. The equations used in the functions block are the following:

Non-saturated conditions:

$$\begin{cases} i_j = i_j^{ref} \\ V_j = V_j^{ref} \\ V_n = 0 \end{cases}$$

One phase saturation (phase *a*):

$$\begin{cases} i_{b,c} = i_{b,c}^{ref} \\ i_a = -(i_b + i_c) \\ V_a = V_s \cdot \text{sign}(V_a^{ref}) \\ V_0 = V_a - R_a i_a - E_a \\ V_{b,c} = V_0 + R_{b,c} i_{b,c} + E_{b,c} \end{cases}$$

Two phases saturation (phases *a, b*):

$$\begin{cases} i_c = i_c^{ref} \\ V_{a,b} = V_s \cdot \text{sign}(V_{a,b}^{ref}) \\ R^* = R_a R_b / (R_a + R_b) \\ i_{a,b}^* = (V_{a,b} - E_{a,b}) / R_{a,b} \\ V_0 = R^* \Sigma_j i_j \\ V_c = V_0 + R_c i_c + E_c \end{cases}$$

All phases saturation:

$$\begin{cases} V_j = V_s \cdot \mathrm{sign}(V_j^{ref}) \\ R_{eq}^{*} = \Pi_j(R_j)/(R_aR_b + R_aR_c + R_bR_c) \\ i_j^{*} = (V_j - E_j)/R_j \\ V_0 = R_{eq}^{*}\Sigma_j(i_j^{*}) \\ i_j = (V_j - V_0 - E_j)/R_j \end{cases}$$

The symbols used are: V^{ref} reference voltage, $V_{a,b,c}$ phase voltages, V_0 neutral voltage, i_{ref} reference currents, $i_{a,b,c}$ phase currents, R_j phase resistances, $E_{a,b,c}$ back-EMFs. R^* and i^* are non-physical quantities used to shorten the equations.

The *Case eval* truth tables simply compares the stage calculated voltages to supply voltage, and if detects a threshold crossing it will consider that phase as saturated and it will provide the appropriate flag in accord to the truth table shown in Fig. 3.21. This flag, and the one provided by either *PMSM Model 1st stage* or *PMSM Model 2nd stage*, provides the new saturation conditions to be used as case selection in the following stage.

This implies that the number of phase which are voltage-saturated can only remain constant or increase from one stage to the next. This approach is not optimal since for every condition three passes are done anyway, which is wasteful. If all three phases are non-saturated or saturated in a single stage, then additional evaluation will not yield different results and can safely be skipped. Such optimization will be implemented in further versions of the model.

Current transfer function

Up to this point, the model is instantaneous, since only the effect of the resistance is considered and the phase inductance is set to zero. Since this is a strong approximation, the effect of the inductance is modeled in this subsystem.

Starting from the physics of the system, the inductive effect has been implemented (Fig. 3.22) using a firts-order low-pass filter on the current, in the form of:

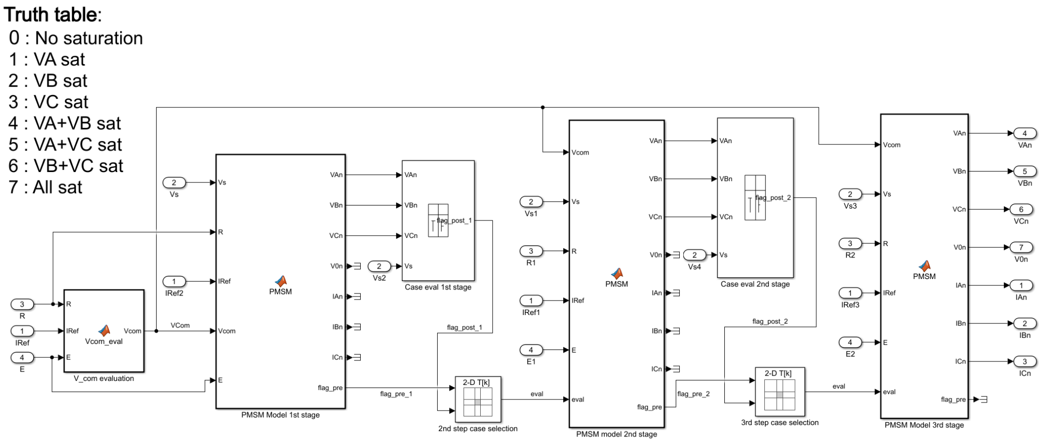


Fig. 3.21 MF PMSM subsystem

$$H_j(s) = \frac{1}{k_t \frac{L_j}{R_j} \cdot s + 1} = \frac{1}{k_t \cdot \tau_{LR} \cdot s + 1} \tag{3.9}$$

where $H(s)$ is the transfer function, L_j and R_j the j-th phase inductance and resistance and k_t is a corrective constant, which has been empirically set to 0.25 to obtain the best match between the HF and MF model. The ratio L/R has been selected as a starting point since it represents the characteristic time of the motor electrical response.

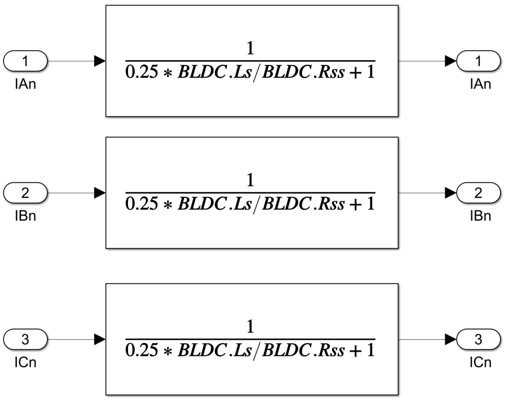

1 IAn
1
0.25 ∗ BLDC.Ls/BLDC.Rss + 1
1 IAn
2 IBn
1
0.25 ∗ BLDC.Ls/BLDC.Rss + 1
2 IBn
3 ICn
1
0.25 ∗ BLDC.Ls/BLDC.Rss + 1
3 ICn

Fig. 3.22 MF current transfer function

3.4 Ehnanced Low Fidelity (ELF) PMSM model

Starting from the medium fidelity model, additional simplification can be performed, again focusing on the slowest subsystems and those that are not fundamental in a monitor model. Model overview is shown in Figure 3.23; once again, the structure of the model is very similar to previous, higher-fidelity versions.

The main difference to the previous model is the implementation of the electric behavior of the stator, which is now implemented using a linearized, saturated form; three phases structure is maintained.

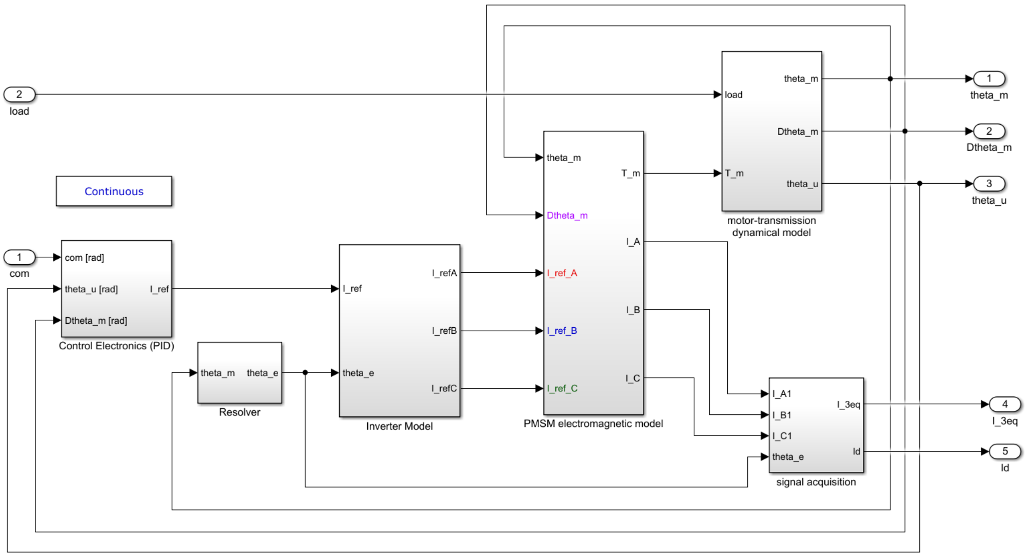


Fig. 3.23 ELF model overview

Most subsystems are the same used in the MF model, as code recycling was one of the objective of the development. For example, the inverter model shown in Fig. 3.24 is the same as that shown in Fig. 3.19 relative to the MF model; both models directly use the reference currents to drive the motor.

3.4.1 ELF PMSM electromagnetic model

The main peculiarity of the PMSM model can be found in the PMSM model subsystem (Fig. 3.25), and in particular in the electrical model implemented (Fig. 3.26). Most subsystems are identical to the previous, for example the *computation of back-EMF coefficients*, *Back-EMF calculation* and *Computation of motor torque*.

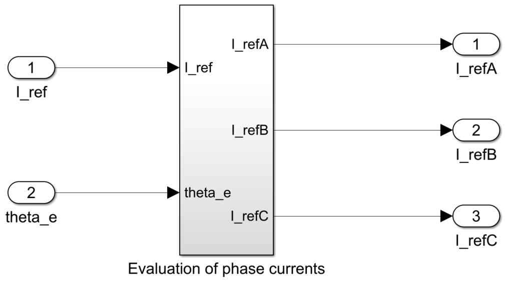


Fig. 3.24 ELF inverter subsystem

On the other hand, the electrical model is much simplified and assumes the form of three linear, saturated responses for the three phases currents. In this sense, the model is a purely ohmic model since no inductive component is considered. The model has thus zero response time, which is again an approximation of the real system.

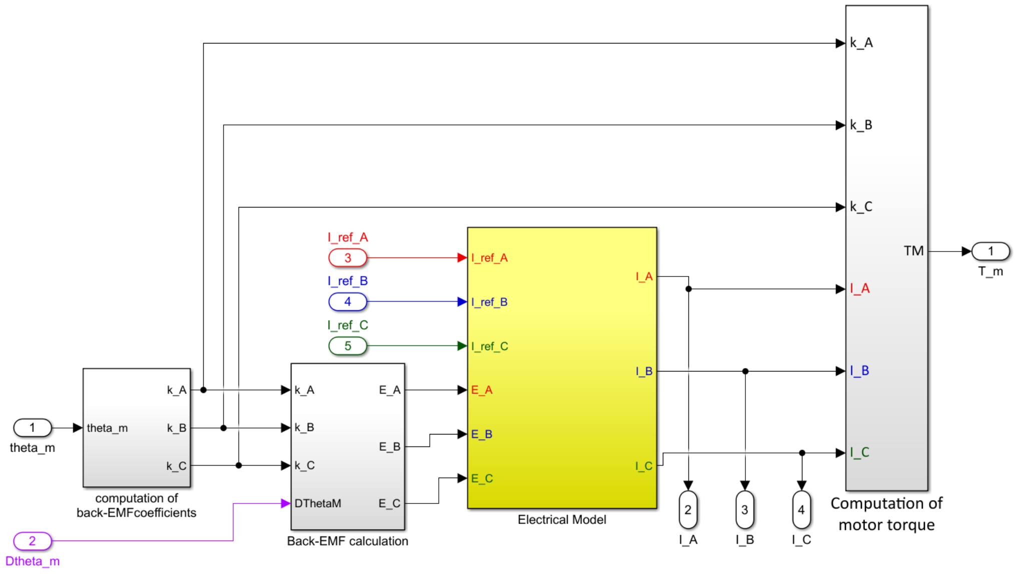


Fig. 3.25 ELF PMSM subsystem

Inside the three *Corr I_j* subsystems, the evaluation of the current produced by each motor is carried out, as shown in Fig. 3.27. The model implemented is simple and can be expressed in mathematical form as:

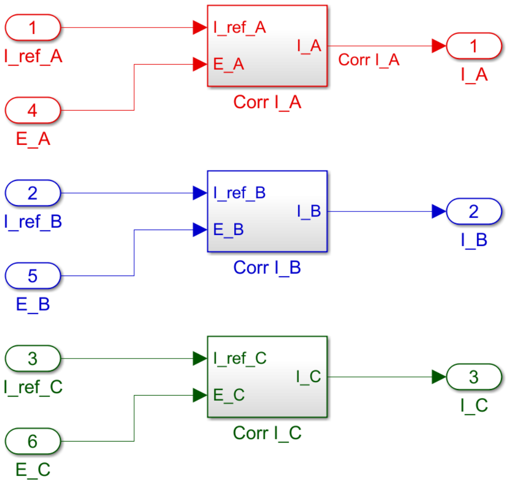


Fig. 3.26 ELF electrical model

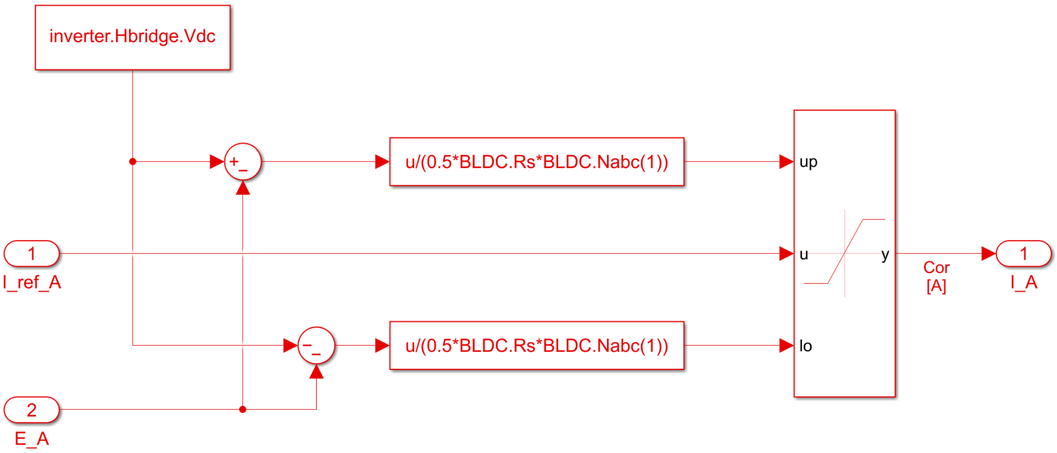


Fig. 3.27 ELF current saturation model (phase A)

$$I_j = \begin{cases} I_{max,j} = \dfrac{V_{dc} - E_j}{0.5 \cdot R_s \cdot N_{abc}(j)} & , \quad I_{ref,j} \geq I_{max,j} \\ I_{ref,j} & , \quad -I_{max,j} \leq I_{ref,j} \leq I_{max,j} \\ -I_{max,j} = \dfrac{-V_{dc} - E_j}{0.5 \cdot R_s \cdot N_{abc}(j)} & , \quad I_{ref,j} \leq -I_{max,j} \end{cases} \tag{3.10}$$

where I_j is the actual phase current, $I_{ref,j}$ is the phase reference current calculated by the controller, $0.5 \cdot R_s \cdot N_{abc}(j)$ is the effective resistance of the considered phase calculated by multiplying the nominal phase resistance $(0.5 \cdot R_s)$ times the percentage of active turns ($N_{abc}(j)$), E_j is the phase back-EMF voltage, V_{dc} is the supply voltage and $I_{max,j}$ is the maximum current output given the operating conditions (speed, supply voltage, fault).

3.5 BLDC High Fidelity model

Before PMSM models were developed, given the availability of a test bench, simpler BLDC modes have been created to provide a basic framework for more complex developments. Furthermore, these models have been used to test some FDI algorithms as it will be shown in later chapters.

The first development has been an high-fidelity, component level model of a Brushless DC driven electromechanical actuator, as shown in Figure 3.28.

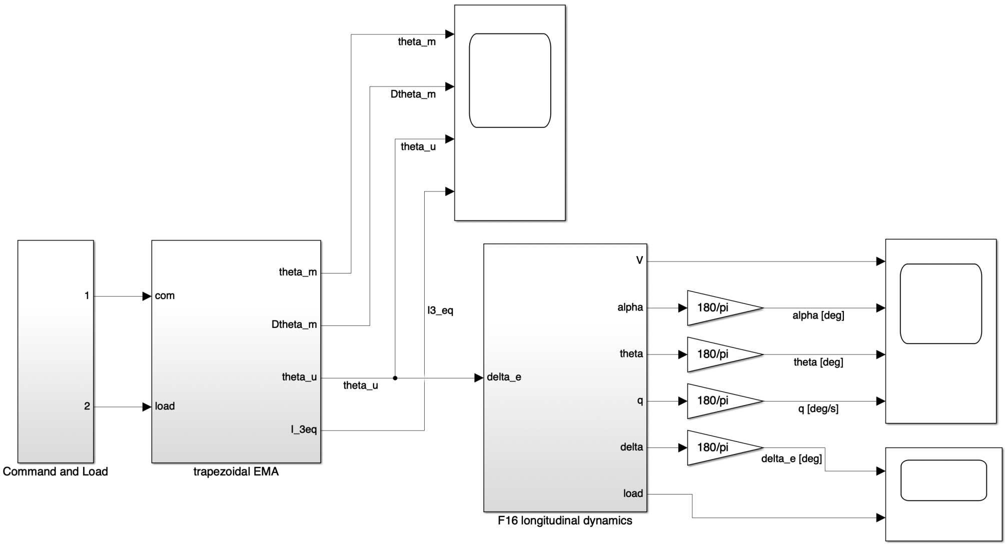


Fig. 3.28 BLDC HF overview

Overall architecture is very similar to previous model; starting from a *Command and Load* subsystem where a reference position signal and fixed external load are set, the two values*com, load* are then fed to the *trapezoidal EMA* subsystem, shown in Figure 3.29, which models the BLDC motor electromagnetic interaction and evaluates the produced motor torque in slow shaft (thus including the reducer); finally, the calculated user position is used to evaluate the aerodynamic action on an aircraft model, modeled using state-space representation. The EMA is supposed to actuate a primary flight control of a fighter jet, namely an F16, and thus affecting its longitudinal dynamics.

The external load obtained by the state space representation can be used in a feedback loop inside the *trapezoidal EMA* subsystem to model actuation during operations, or it can be set externally to an arbitrary value, as in Fig. 3.28 to simulate ground operations.

3.5.1 Trapezoidal EMA subsystem

The *trapezoidal EMA* subsystem has a structure almost identical to that of the PMSM variant, as visible from Figure 3.29. Only the main differences will be described in the following.

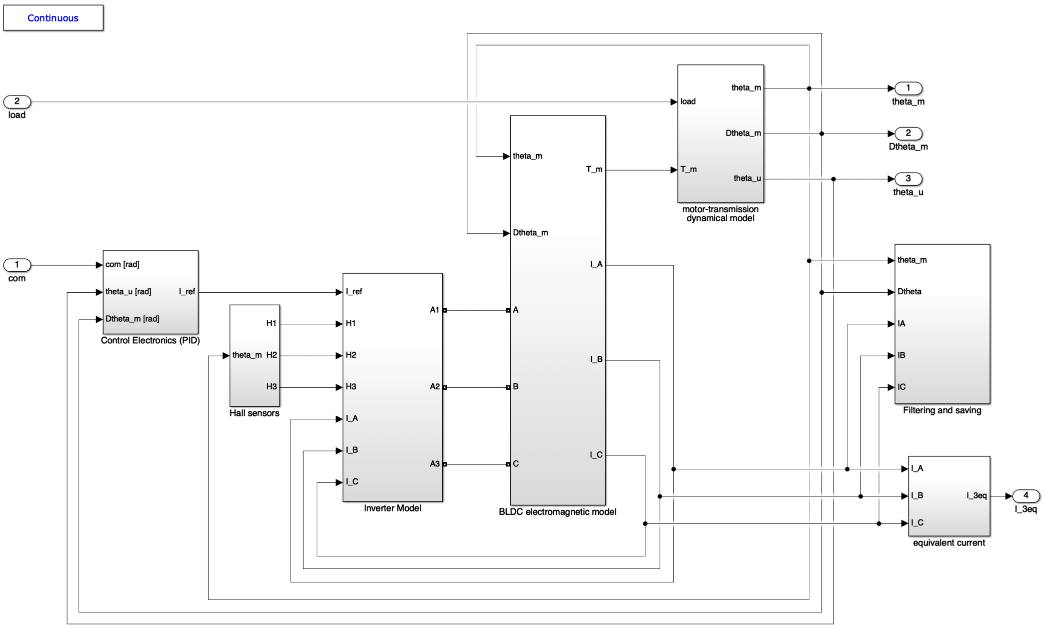


Fig. 3.29 BLDC trapezoidal EMA subsystem

Control electronic subsystem

The *control electronic (PID)* subsystem is very similar to that of the PMSM, as this last was directly derived from this one. Once again, the outer position control loop has an additional control loop inside, for motor angular velocity.

After being saturated, the speed error is passed through a PID block, which creates a reference torque (*TM_ref*) which is scaled by a gain (indicated with -K-), with a value of $1/BLDC.KE/2$, i.e. the back-emf coefficient of a single phase.

This calculates the needed to current to achieve that torque in the current speed configuration; the current is then saturated to simulate the maximum allowable motor current.

Finally, the *I_ref* is the reference current which is used to drive the motor.

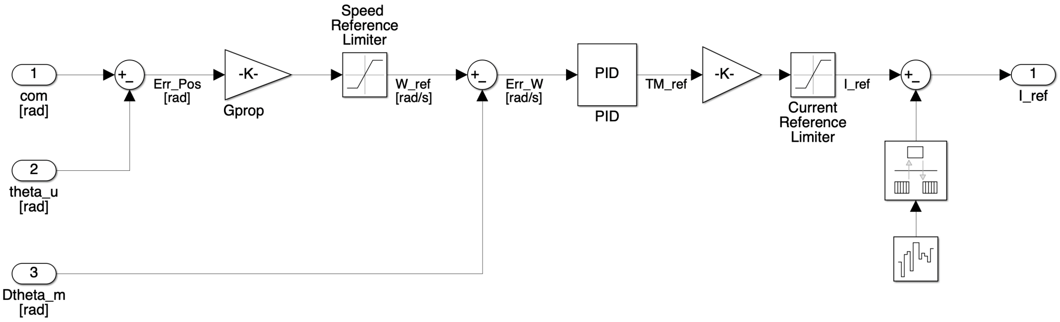


Fig. 3.30 Control electronic subsystem

Looking inside the PID block (Fig.3.31), the implementation is very straightforward and has a parallel architecture, with three gains in explicit form. The derivative part is filtered to remove high frequency noise on the derivative signal.

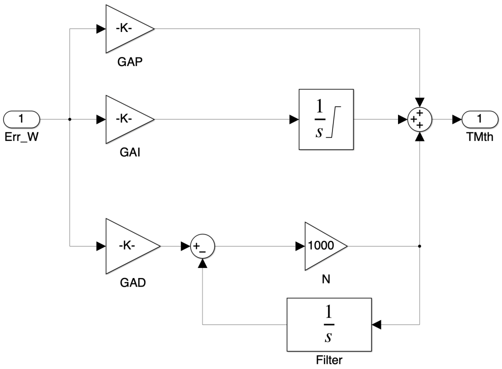


Fig. 3.31 Control electronic subsystem, PID block

3.5.2 Hall sensors subsystem

In the *Halls sensor* subsystem, differences are visible when compared to the PMSM model, as the computation of the motor angle is not evaluated in this block but rather inside the next subsystem (*Inverter model*).

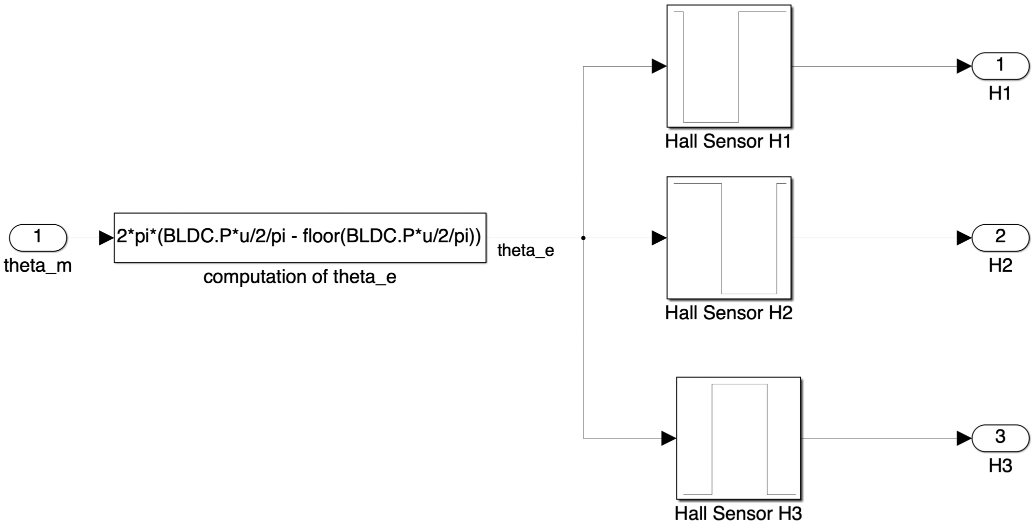


Fig. 3.32 Halls sensor subsystem

In Figure 3.32, the input signal is the motor position, *theta_m*, which is used to compute the electrical angle - *theta_e* by multiplying by the number of poles *BLDC.P* and wrapping to 2π as usual.

The electrical angle is used to selectively activate the three Hall sensor blocks which turns on or deactivate depending on the electrical position. These sensors allow the controller to evaluate the angular position of the motor to allow a correct commutation scheme. They are modeled by using logical tables depicting a square wave which is properly shifted for each of the three sensors.

The three logical signals are then fed to the next subsystem, *Inverter model*, to compute the correct switching sequence and modify the reference currents to apply to the three phases.

3.5.3 Inverter model

The inverter model accomplishes several function with each of his subsystems: firstly, it evaluates the active phases given the motor current position, obtained through the three logical Halls sensor signals; it then uses this information to activate or deactivate the phase by multiplication of the *I_ref* times the logical output (0 or 1), as shown in Figure 3.34.

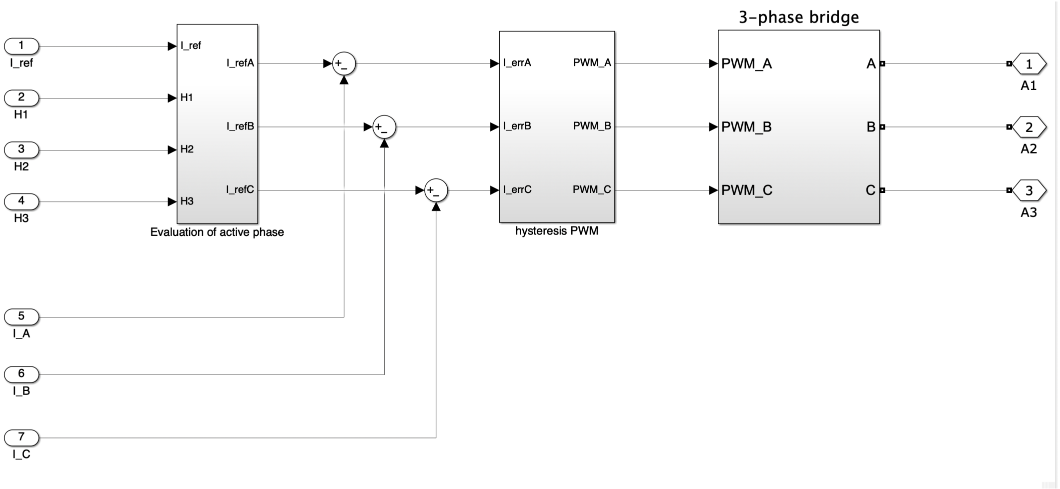


Fig. 3.33 Inverter model subsystem

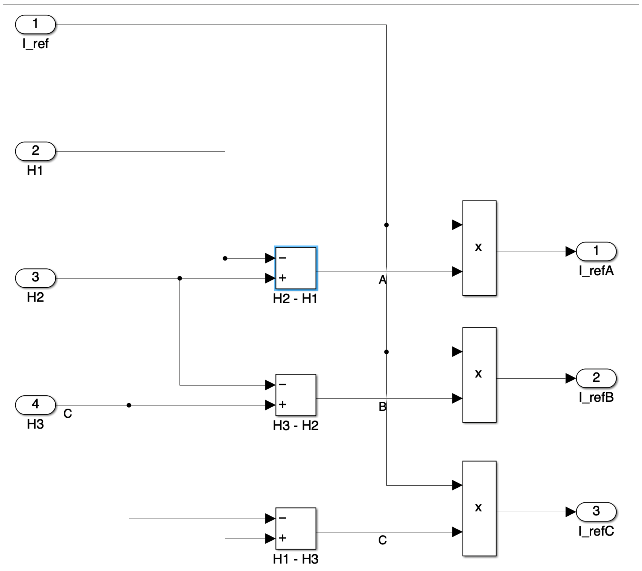


Fig. 3.34 Active phase subsystem

Once the three individual reference signals have been determined ($I\_ref_{A,B,C}$), the measured value is subtracted ($I\_{A,B,C}$) to obtain the error-related currents to be applied to each phase.

Then, the same PWM shown in Figure 3.10 is used. It is based on an hysteresis band which turns on and off the driver MOSFETS according to the hysteresis band conditions.

Finally, the 3-phase bridge is the same shown in Fig. 3.11 which uses Simscape Power Systems elements to model a 6-transistors H-bridge configuration; it is driven by 6 signals - 3 direct coming from the *hysteresis PWM* subsystem and the 3 negations.

After that, the energy connections (hexagonal ports) are connected to the *BLDC electromagnetic model* which uses them to simulate the electromagnetic interaction between stator and rotor.

3.5.4 BLDC electromagnetic model

This subsystem models the electromagnetic interaction caused by the current flow in the stator phases and the rotor magnetic field. As per the PMSM case, this is a simplified approach which does not models reality to the component level, but rather uses an approximation.

In Figure 3.35 the overall subsystem is shown, and it can be noted it is very similar to that of the PMSM - Fig. 3.12. However, the main difference is found inside the *computation of back-EMF coefficient* subsystem.

As shown in Figure 3.36, the way in back-EMF coefficients are calculated is different from the PMSM case. In fact, one of the main differences between the two types of motors is the shape of tha back-EMF curve; while it is a sinusoid for the PMSM, it tends to a square (or trapezoidal) wave in the case of the BLDC.

The three f(u) blocks are simple modulation functions that takes into account the effect of partial phase shorts and static eccentricity on the back-EMF coefficient curve.

For example, for the phase A we have:

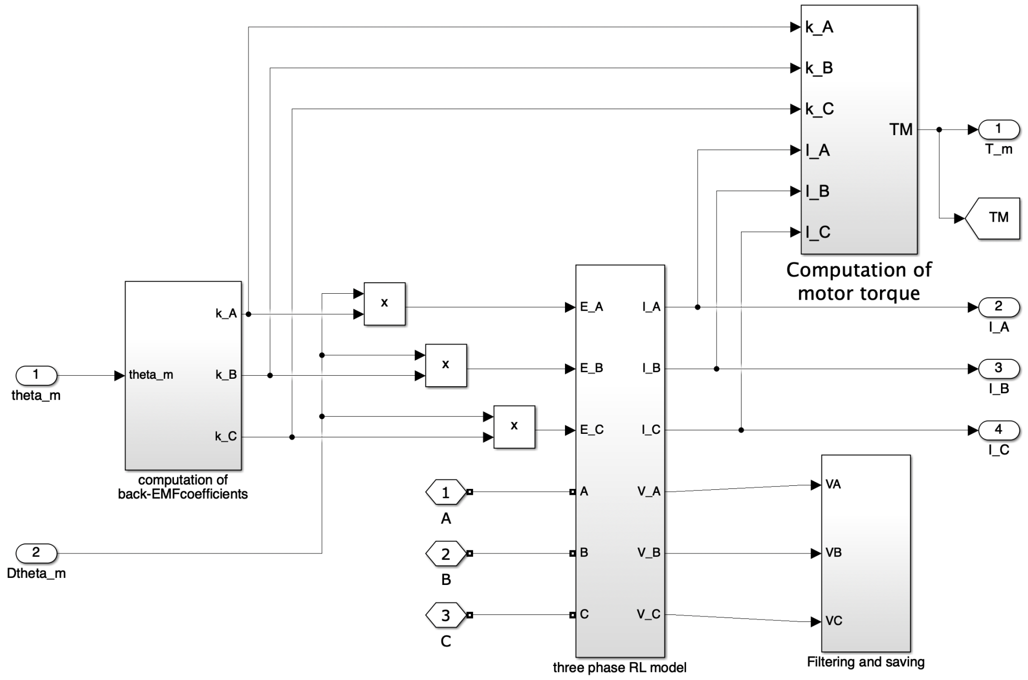


Fig. 3.35 BLDC electromagnetic model

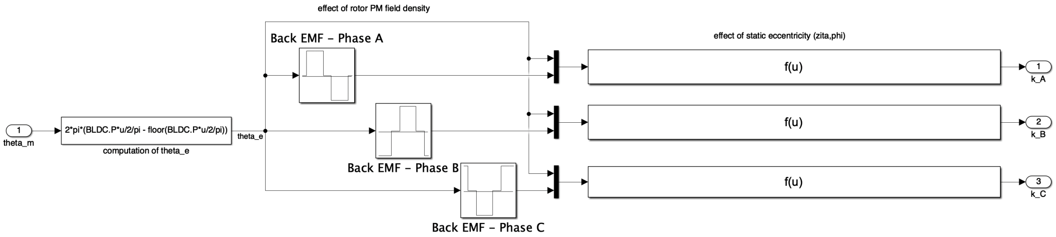

effect of rotor PM field density
Back EMF - Phase A
Back EMF - Phase B
Back EMF - Phase C
effect of static eccentricity (zita,phi)
1
theta_m
2*pi*(BLDC.P*u/2/pi - floor(BLDC.P*u/2/pi))
computation of theta_e
theta_e
f(u)
f(u)
f(u)
1
k_A
2
k_B
3
k_C

Fig. 3.36 Back-EMF coefficient calculation subsystem

$$BLDC.Nabc(1)*BLDC.Ke*u(2)*$$
$$*(1+(BLDC.P>1)*BLDC.zeta*cos(u(1)-BLDC.phi))$$

where $u(1)$ is the electrical angle θ_e and $u(2)$ is the output of the table relative to phase A that models the square waveform of the back-EMF coefficient as function of the angular position. Additionally, $BLDC.Ke$ is the back-EMF coefficient (nominal), $BLDC.P$ is the number of pole pairs, $BLDC.Nabc(1)$ is the usual percentage of shorted turns relative to stator phase A and *BLDC.zeta* and *BLDC.phi* are respectively eccentricity amplitude and phase.

The other subsystems are the same as those of the PMSM and will not be described again; the equal subsystems are the *three-phase RL model* (same as in Fig. 3.14) and *Computation of motor torque* - which is the same as that in PMSM case (see Eq. 3.7).

The *Filtering and saving* subsystem uses the same third-order low-pass filter to remove high-frequency switching noise from the voltages and has the same structure as that of Fig. 3.17.

Other subsystems

The other subsystems shown in Fig. 3.29 are identical to those described in the PMSM section, and will thus not be reported again. In particular, the *motor-transmission dynamical model* uses the same Borello-based dry friction model, and includes once again viscous friction.

The *Filtering and saving* subsystems simply uses rate transition blocks to sample all the quantities of interest with the same frequency and uses 'To Workspace' blocks to save said data to the Matlab workspace.

Finally, the *equivalent current* subsystems applies the following formula to evaluate the single-phase equivalent current:

$$i_{3,eq} = \frac{1}{2}\sum(|i_A| + |i_b| + |i_c|) \cdot sign(T_m) \tag{3.11}$$

where T_m is the motor torque. This simple formulation stems from the fact that in a BLDC motor only two phases are active at the same time, and summing the absolute values yields double the actual value of the equivalent current. For example, if phase A and B are active, one would find that $i_C = -i_A$ while $i_B = 0$, thus the equivalent single-phase current would be $|i_C| = |i_A|$. Finally, since the absolute value removes the sign to the currents, the sign is obtained by considering the sign of the motor torque.

3.5.5 F16 longitudinal dynamics

The other important, and different, subsytem present in the model is a simplified, state-space representation, of the longitudinal dynamics of an F16 aircraft as shown in Fig. 3.37. The values of the parameters are directly taken from [50].

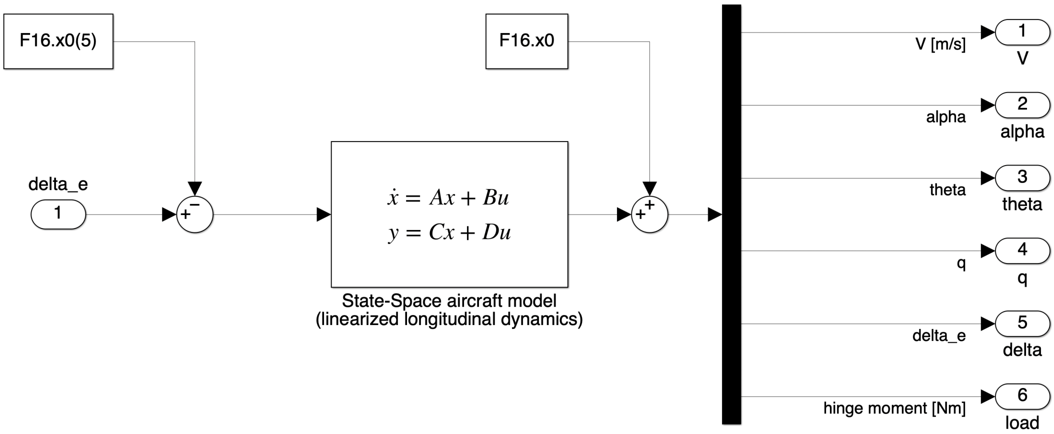


Fig. 3.37 Longitudinal dynamics subsystem

This a simplified, linearized form to describe the longitudinal dynamics of an aircraft. It stems directly from the equation of motion, which are then linearized and written in matrix form. *delta_e*, which is the input of the subsystem, is the elevator deflection, as the whole Simulink model a primary flight control actuator. Then, the element *F16.x0(5)* is subtracted, since it is the initial condition (or trim). The initial conditions vector includes speed *V*, angle of attack *alpha*, the horizontal body angle *theta*, the pitch rate *q*, elevator deflection *delta_e* and finally hinge moment (*load*).

The hinge moment can be used as input for the whole EMA subsystem to simulate a generic flight condition and actuation.

3.6 BLDC Low Fidelity model

For some application, the detailed component modeling of the electrical part of the EMA is not only unnecessary, but it could also be counterproductive. In fact, the electrical subsystem is the slowest subsystem to simulate, given the microsecond timescale of the phenomena. This strongly limits the maximum allowable timestep to values in the order of 10^{-6} s, making simulations very slow and very memory-heavy.

One such example can be found in later chapters where FDI algorithms are presented, focusing on the mechanical part of the system. In this case, the electrical part is not of interest and can (and should) be simplified to greatly reduce computation time.

The overall model scheme is shown in Fig. 3.38, from [51].

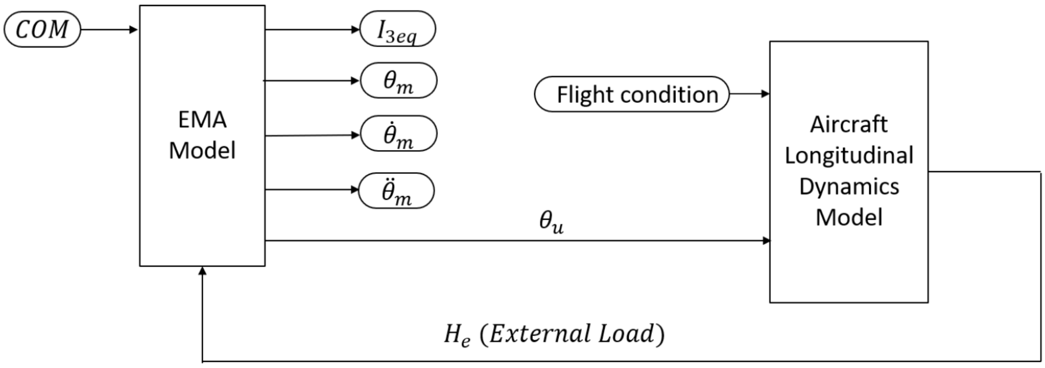


Fig. 3.38 Low Fidelity BLDC model overview

However, in this case, the model does not have an externally imposed external load, rather it uses the dynamics model of an aircraft to calculate it. Thus, the model does not represent a bench mounted EMA, but is representative of a primary actuation system - particularly of an F16 elevator, as previously exlained.

The EMA model subsystem is shown in Fig. 3.39, and it is composed by three elements: *Controller*, *Electrical model* and *Mechanical model*.

The *Controller* is a simplified control system made of two, nested, feedback loops, as shown in Fig. 3.40. The commanded position is compared against the actual position to evaluate the position error, which is then multiplied by the first gain block to obtain a speed reference, which is then saturated.

This value is compared against the actual motor speed to obtain a speed error, which is then multiplied by the second gain block to obtain the reference current, after saturation.

Using multiple cascaded control loops is useful in creasing the stability margin of a system, allowing a more rapid response to changing conditions and reducing the effects of delays.

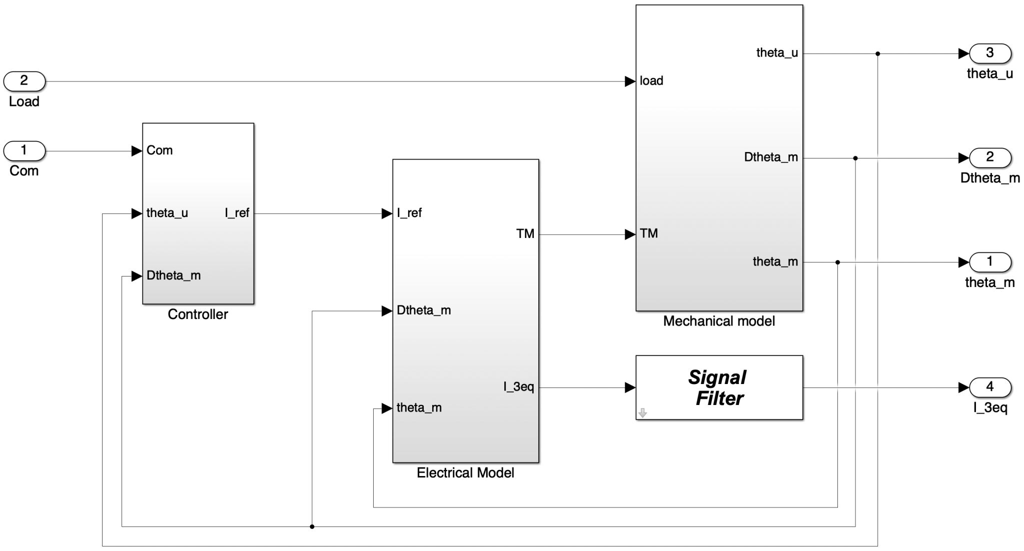


Fig. 3.39 Low Fidelity BLDC EMA subsystem

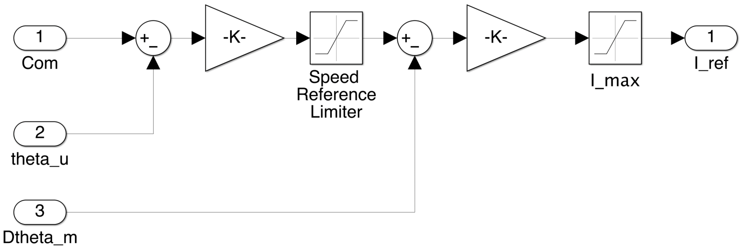


Fig. 3.40 Low Fidelity BLDC Contoller

Inside the electrical subsystem, shown in Fig. 3.41, a greatly simplified version of the electrical part of the EMA can be found. Rather than focusing on a component level interaction with physical links, as in 3.35, in this case the whole approach deviates from reality and uses some shape functions to emulate the effect of eccentricity and partial short-circuit, in an analogous way as the HF model.

In this subsystem, an additional, internal loop on current is found - which is basically a control loop on acceleration. After calculation of the current error, a bang-bang control activation scheme is used. In this case, the sign of the current error is used to power up the motor with either +Vdc or -Vdc. The bang-bang scheme is a simplified version of the hysteresis control implemented in the HF-model, and is much faster to compute since there is no deadzone nor zero-crossings to process.

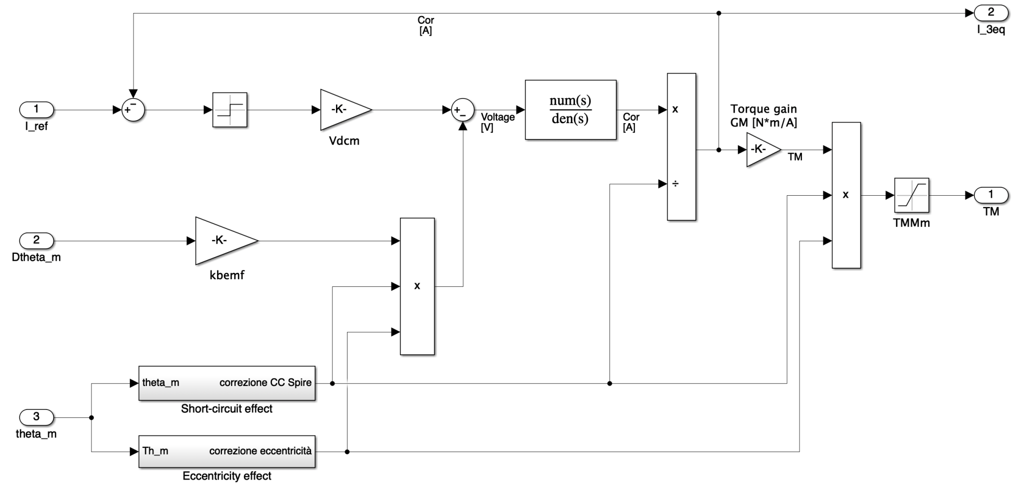


Fig. 3.41 Low Fidelity BLDC electrical subsystem

The motor characteristic is implemented by a simple transfer function, and the motor has both resistive and inductive components.

Finally, the back-EMF is calculated in the usual way by multiplication of the motor speed by the back-EMF coefficient. The coefficient is modulated by two shape functions, shown in Fig. 3.42 and Fig. 3.43, which are derived from physics in the case of the short-circuit, or from best-fit curves in the case of the eccentricity effect - starting from the HF model as reference.

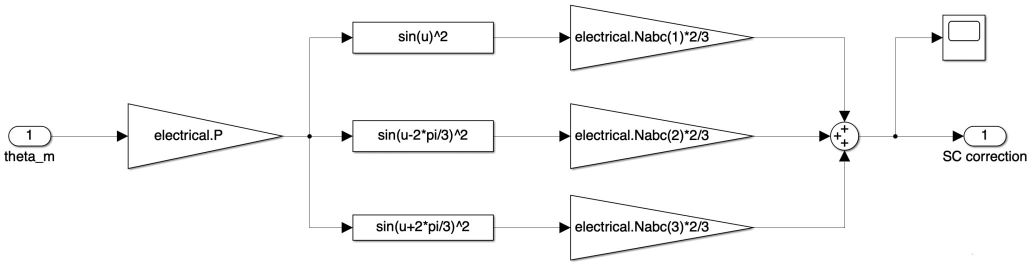


Fig. 3.42 Short circuit back-emf coefficient correction

On the other hand, the *mechanical model* has been ported directly from the HF model and is shown in Fig. 3.44.

Finally, the longitudinal dynamics subsystem used to evaluate the hinge moment is the same of the HF model shown in Fig. 3.37.

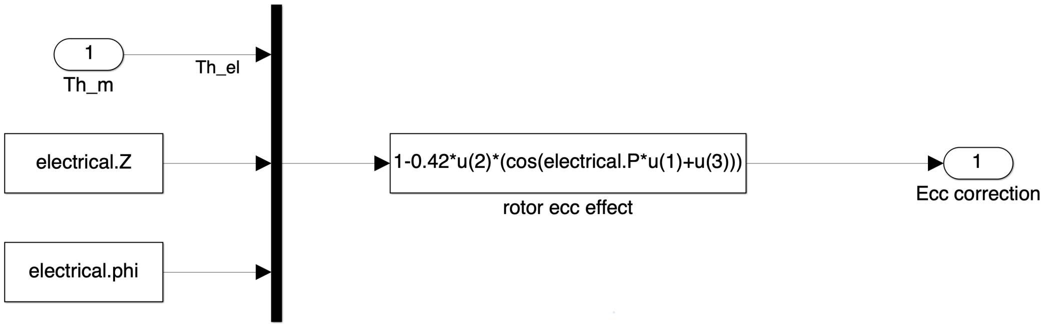


Fig. 3.43 Eccentricity back-emf coefficient correction

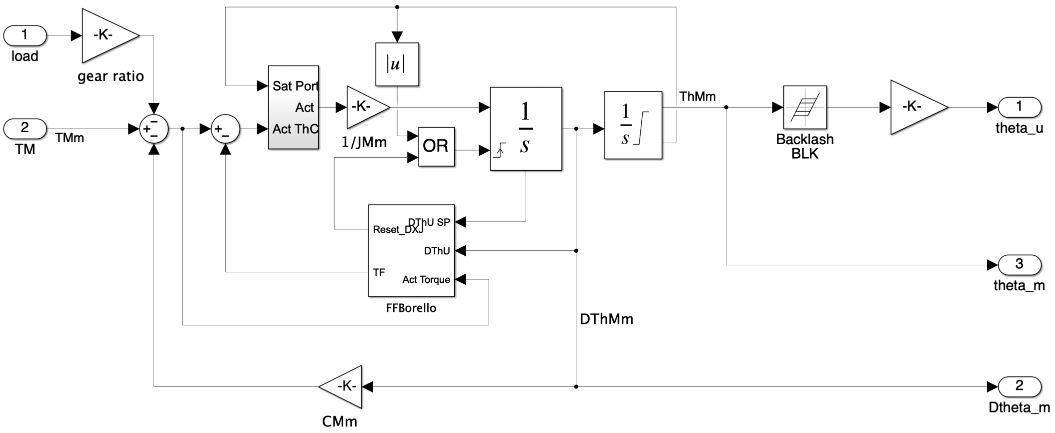


Fig. 3.44 Low Fidelity BLDC mechanical subsystem

Chapter 4

Fault Detection and Identification algorithms

Fault detection and identification (FDI) is a branch of control engineering that focuses on monitoring a system to determine when and where a fault has occurred. The objective of this branch is to prevent faults and to ensure that systems can operate in a safe and reliable way. Methods used for fault detection varies depending on data availability and sources.

In FDI, two approaches are usually adopted: model-based FDI and signal processing based FDI. In model-based FDI, a model of the system is used to determine if a fault has occurred. Models are usually mathematical description of the behavior of the system, with varying assumptions made.

On the other hand, signal processing based FDI leverages mathematical or statistical operations, which are performed on measurements using a neural network or similar machine learning tools, used to extract information about the fault.

Fault detection is achieved by generating fault indicator signals called residuals and comparing them to a threshold, which is set depending on the task. Identification is more complex since in real systems multiple faults can appear at the same time and some faults combinations can influence each other.

4.1 Framework of operation

The models described in the previous chapter are useful for testing new Fault and Identification (FDI) algorithms, which is something not feasible on a real system for several reasons, such as cost, since many system have to be tested to obtain a meaningful dataset, and complexity, since creating a level of damage accurately is hard for most faults. Additionally, since several faults are considered at the same time, the dimensionality of the problem greatly increases and thus the required dataset size.

In this chapter, several implementations of FDI techniques, based on hybrid approaches will be described. Application case will be the electromechanical actuator described in the previous chapter, and FDI will aim at detecting and evaluating faults from both the mechanical and the electrical domains.

All the techniques that will be described follow the scheme shown in Fig. 4.1, in particular focusing on the offline component. In fact, to train the neural network, which is the fault evaluator, a large dataset of operating data is needed. Additionally, the data will need to be labeled, i.e. the exact amount of damage present in each simulation must be known. Thus, the utility of the models presented in the previous chapter, which can be seeded with exact faults and simulated to provide a realistic representation of the physical system.

The dashed boxes in the figure represents further developments of FDI, namely Remaining Useful Life estimation and prognostics, which will not be covered in this thesis.

4.2 Electrical motor and shaft eccentricity FDI

The first application of the hybrid, machine-learning assisted FDI has been the detection and quantification of the damages related to the electrical motor, such as partial phase shorts. Additionally, rotor static eccentricity is modeled and included in the diagnostic process.

The model used to evaluate the system has been a modified HF model, where in this case a brushless DC motor (BLDC) has been used, rather than a PMSM which was used in the previous chapter. However, functional architecture is the same and

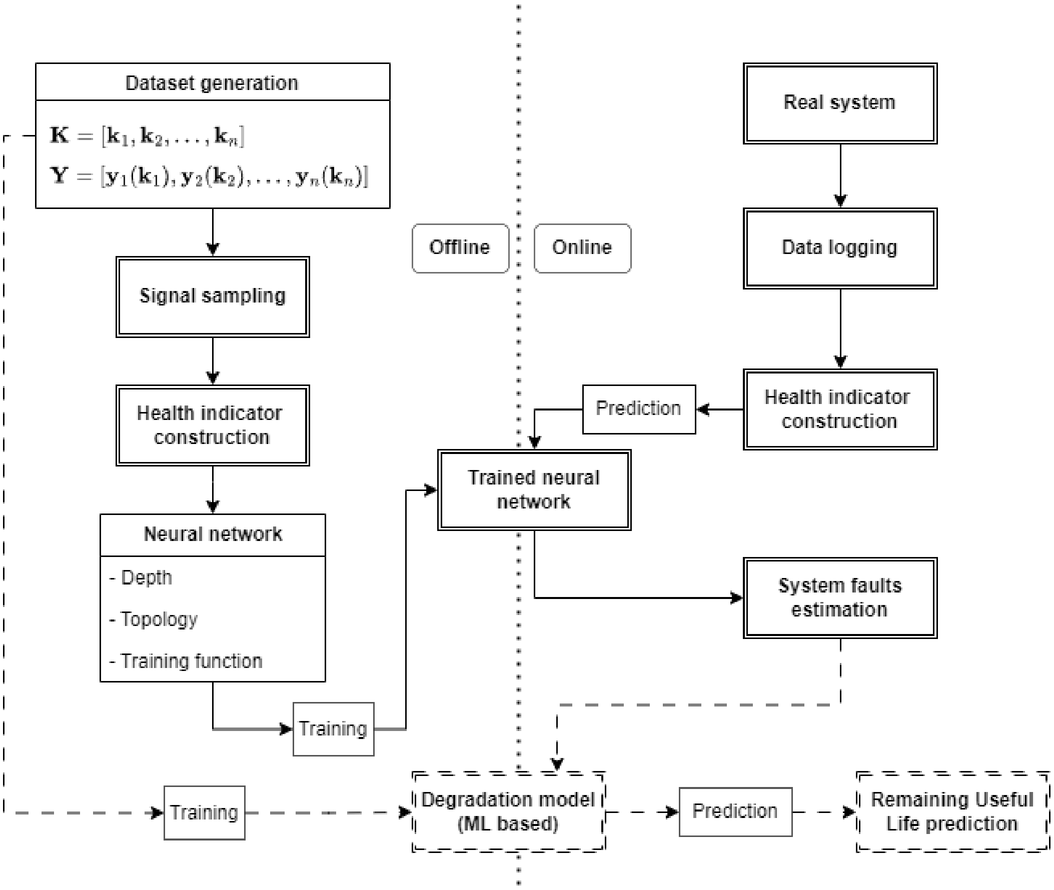

Dataset generation
$\mathbf{K} = [\mathbf{k}_1, \mathbf{k}_2, \ldots, \mathbf{k}_n]$
$\mathbf{Y} = [\mathbf{y}_1(\mathbf{k}_1), \mathbf{y}_2(\mathbf{k}_2), \ldots, \mathbf{y}_n(\mathbf{k}_n)]$
Signal sampling
Health indicator construction
Neural network
- Depth
- Topology
- Training function
Training
Offline
Online
Real system
Data logging
Health indicator construction
Prediction
Trained neural network
System faults estimation
Training
Degradation model (ML based)
Prediction
Remaining Useful Life prediction

Fig. 4.1 Algorithm overview

can be found in [52], where this technique has been initially presented. The main difference between a PMSM and a BLDC is that while a PMSM has a sinsuoidal back-EMF profile, BLDCs presents a trapezoidal profile. This leads to a difference in driving logic, which is much easier fort BLDCs.

Referring to Figure 4.1, **k** is the generic fault vector, defined as the vector which contains information regarding the faults present in the system, while $\mathbf{Y}(\mathbf{k})$ is the system response when the fault vector is applied to it.

For this particular application, **k** is defined as:

$$\mathbf{k} = [N_A, N_B, N_C, \zeta, \phi] \tag{4.1}$$

where $N_{A,B,C}$ are the short percentage relating to each phase, with values between 0 (nominal conditions) and 1 (complete short); ζ is the static eccentricity defined as

$\zeta = x_0/g_0$, where x_0 is axis offset from center and g_0 is the nominal air gap; ϕ is the phase of said eccentricity.

Two equations are set as boundary, in order to generate a dataset where the system is still functioning, and also to condense all electrical faults in a single condition. The equations are:

$$\sqrt{N_A^2+N_B^2+N_C^2} \leq 0.5 \tag{4.2}$$

$$\zeta \leq 0.5 \tag{4.3}$$

It has to be noted that the previous boundaries do not represent specific physical conditions, but are rather arbitrary values. Specific values can be obtained considering, for example, performance requirements and thus chosen accordingly. In any case, the algorithm is the same and can be generalized easily to other conditions.

After creating a 5-dimensional randomized matrix, with 3000 vectors, the model is simulated with every fault vector, and data such as current, voltages, angular position and angular speed are logged.

4.2.1 Health indicator

The next step is the construction of an appropriate health indicator, i.e. a signal which is sensitive to faults while being unaffected, to the extent possible, by operating and environment conditions.

For this case, as demonstrated in [53], the back-EMF coefficient is a suitable signal, since it can show partial phase shorts and eccentricity. An example of nominal vs. damaged curve can be found in Fig. 4.2 (from [52]). The orange curve shows a back-emf coefficient curve with periodic behavior (which is the effect off the eccentricity), while the effect of the partial short is not immediately evident but consists in a lowering of the coefficient in angular positions where the phase is active.

The motor considered in the HF model is a three phase motor, however a single-phase equivalent back-EMF coefficient is calculated for simplicity; there is no loss of generality as all the information regarding the three separate coefficients is encoded in the equivalent single-phase one.

The motor constitutive equation is the following:

$$V_j - E_j = V_j - k_{bemf,j}\omega_m = R_m i_j + L_m \frac{di_j}{dt} \tag{4.4}$$

where V_j is the j-th phase voltage; E_j is the back-EMF, calculated as $E_j = k_{bemf,j}\omega_m$, where $k_{bemf,j}$ is the back-emf coefficient for the considered phase and ω_m is motor angular speed; i_j the j-th current and R_m, L_m are nominal phase and resistance. Thus, solving the equation above for each timestep provides three back-EMF coefficients curves as function of time.

In this approach, an assumption is made regarding the resistance and inductance. Both are supposed to be constant, which is a strong assumption since:

$$R_j(N_j) = R_m \cdot N_j \tag{4.5}$$

$$L_j(N_j) = L_j \cdot N_j^2 \tag{4.6}$$

which correlates both resistance and inductance with the amount of partial short present in a single phase. However, this simplified was firstly tested for simplicity and performance were acceptable anyway, so a full algorithm considering these values as variables has not been tested since accuracy was already very high.

Now, after obtaining $k_{bemf,j}(t)$, the curves are resampled to obtain back-emf coefficient as function of the mechanical angle, thus:

$$k_{bemf,j}(\theta_{m,k}) = \frac{1}{n}\sum_{l=1}^{n} k_{cemf,j} \cdot ((\theta_{m,k} - \varepsilon) \leq \theta_m \leq (\theta_{m,k} + \varepsilon)) \tag{4.7}$$

where ε is a small tolerance band.

Now, one has obtained three different back-emf coefficient curves, one for each phase and could use that directly to proceed in the algorithm. However, it is also possible to perform a sort of data compression by switching to the equivalent single-phase motor back-emf coefficient, by simply applying the following:

$$k_{bemf} = \sum_{j=\text{ 1, 2, 3}} |k_{bemf,j}| \tag{4.8}$$

and thus using a single signal which possess information regarding all three phases.

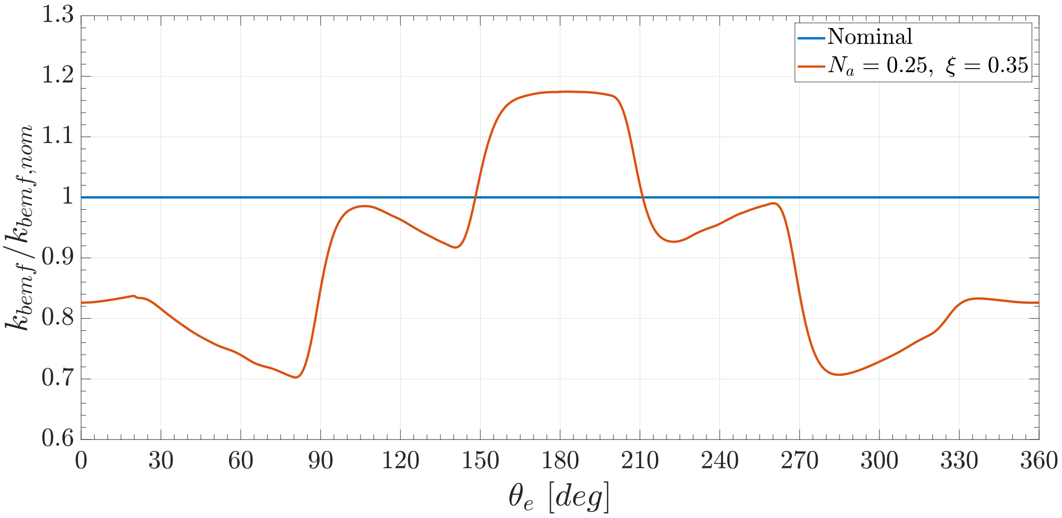


Fig. 4.2 Example of nominal and damaged back-EMF coefficient curve

4.2.2 Sampling

The reconstructed back-EMF coefficient curve showed regularity due to the periodicity of the commutation events that occurred every 30 degrees electrical. This can be seen in Figure 4.2. This regularity could be used to implement various sampling techniques. Three different methods were analyzed, each with a different number of sample points extrapolated for each period between two successive commutations.

The first sampling mode extrapolated the center point value of each intracommutation period, resulting in six points per k_{bemf} curve. The central point was sampled assuming that the commutations were instantaneous and only spanned a single timestep.

The second sampling mode sampled two different points for each intracommutation period, resulting in 12 points per back-EMF coefficient curve. In this case, it was necessary to consider the noise present in the signal. By using linear interpolation, i.e. $y = mx + q$, the maximum angular range where the curve could be approximated as linear before commutation effects occurred was determined. In Figure 4.3 (from [52]), the zone where there was no variation in the angular coefficient of the interpolating line is shown. This was necessary because the k_{bemf} value could vary greatly if sampled near a commutation. An angular amplitude of 20 degrees electrical was chosen, considering a total angular range of 60 degrees electrical between two successive commutations.

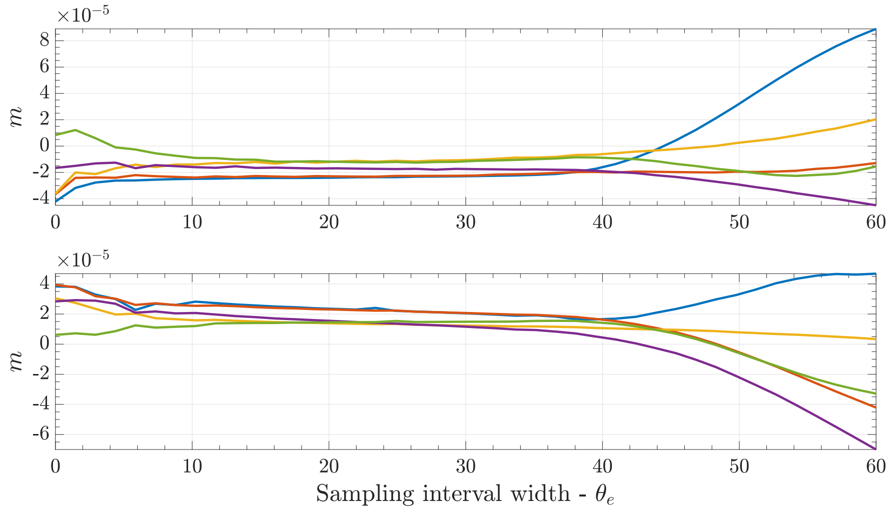


Fig. 4.3 Variation of *m* for two different intra-commutations periods considering 5 randomly seeded back-EMF coefficient curves.

The third method is simply the combination of the two previous methods, i.e. sampling both the midpoint and the two additional equally-spaced samples, for a total of 18 points per curve.

In Figure 4.4 (from [52]) the relevant points (commutations, center points and additional points) are shown for five randomly seeded k_{bemf} curves.

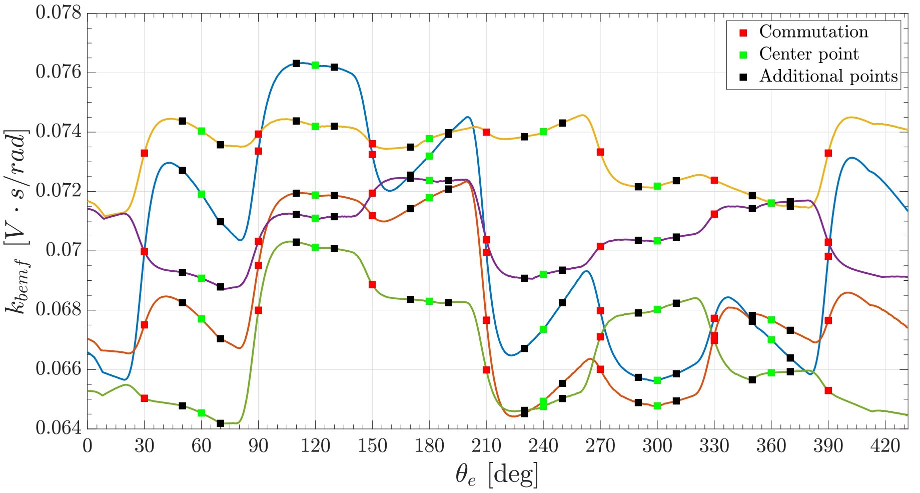


Fig. 4.4 Discretization of five randomly seeded reconstructed back-EMF coefficient curves.

4.2.3 Artificial neural network

Artificial neural networks (ANNs) are a powerful tool for machine learning. They operate on the principle of biological neural networks and learn to perform tasks based on examples without specific programming rules. ANNs can process data that would be difficult to handle using other methods, such as fitting many-dimensional data sets, image classification, and speech recognition.

The structure of a neural network is based on fundamental elements called neurons, which are connected to each other in a way that resembles biological synapses. The combination of neurons and connections defines the topology of the network. Each connection has a weight that represents its importance. These weights are adjusted during training to minimize the error function using a training function, typically with a back-propagation method. There are several learning paradigms, including supervised, unsupervised, and reinforcement learning, each of which is more effective for certain tasks.

Another important characteristic of a neural network is the propagation function, which evaluates neuron inputs based on the outputs of connected predecessor neurons as a weighted sum. A bias term can be included as an additional parameter for each neuron to increase the complexity and capabilities of the network.

In this work, networks were implemented using MATLAB Machine Learning Toolbox. The architecture was either a single layer or two layer perceptron with one or two hidden layers. In general, this class of neural networks is called multi-layer perceptron ([54]), and are very simple networks for today's standards. However, the task at hand is not particularlty difficult, since it is a regression task, albeit on a multi-dimensional dataset.

Networks inputs were the sampled values as previously described, while the outputs were always five - representing the target fault vector. The number of neurons in each hidden layer was changed in size between the number of inputs (18, 12 or 6) and the number of outputs (always 5), thus obtaining a network with a decreasing size between input and outputs. An example of two-layers MLP network can be found in Figure 4.5.

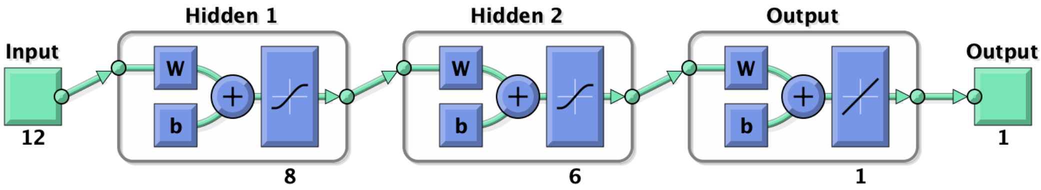


Fig. 4.5 Example of two-layer perceptron

For each architecture, two different learning functions were tested: trainlm, implementing the Levenberg–Marquardt back-propagation algorithm ([55] , [56]), and trainbr, with Bayesian regularization [57].

Performance function for every run was MSE (Mean Square Error), which is usually good fot regression tasks, with a target accuracy of 10^{-6}. The dataset was subdivided, randomly, into three subsets, used for training, validation and testing with ratio $70\%,\ 15\%,\ 15\%$.

The transfer function used for all networks was symmetric saturated linear function, as shown in Figure 4.6; the function output was linear in the $[-1,1]$ region, while it assumed constant values outside the interval (-1 for $x < -1$ and 1 for $x > 1$). This function is easy to compute when compared to hyperbolic tangent, and provides the non-linearity needed to fit non-linear datasets.

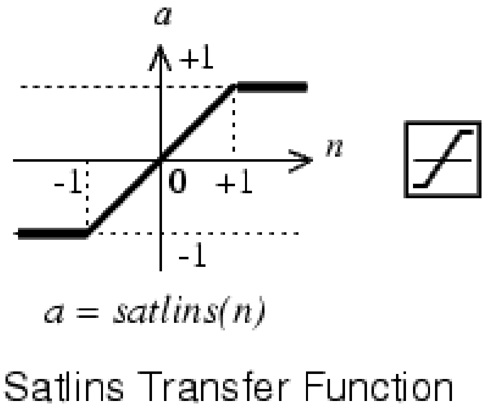


Fig. 4.6 Saturated linear symmetric transfer function.

4.2.4 Results

In the simplest network architecture based on the first sampling mode, i.e. by using a single point per commutation, we have six inputs and a single hidden layer of size 5 or 6 (Figure 4.7, [52]).

Addition of a single neuron increased network performance, even tough there still was low accuracy. The Bayesian regression scored better with five neurons, with very small performance difference with six neurons.

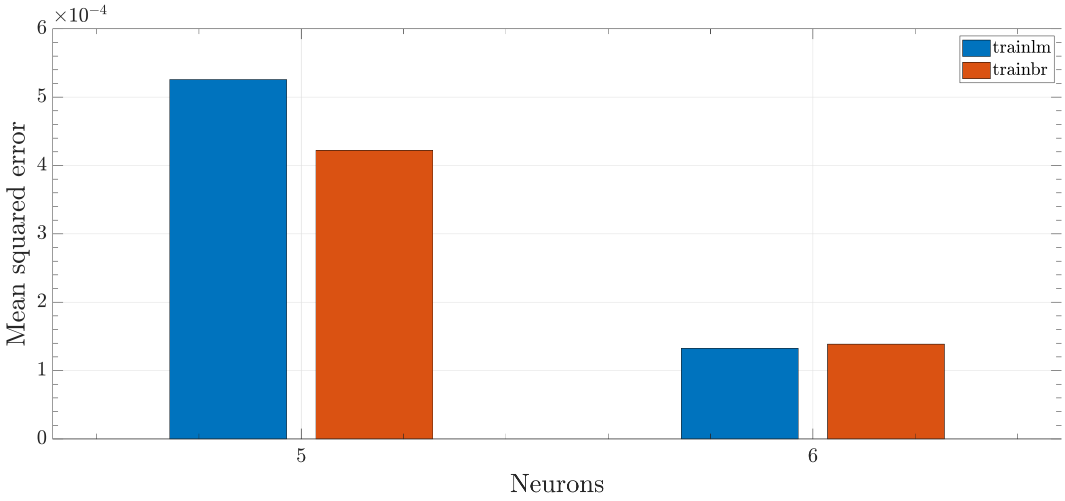


Fig. 4.7 Single hidden layer networks performance as function of neurons and training function, sampling mode 1.

Using a more complex sampling strategy, the number of inputs was increased to two per commutation - 12 in total. In general, increasing the available inputs lead to an increase in prediction accuracy. For this condition, 6, 9 and 11 neurons networks were considered.

As expected, the performance score (which is the mean squared error) decreased with the increase of neurons, to value less than 10^{-5} in the case of 9 or 11 neurons as visible in Figure 4.8 ([52]), achieving better predictions in comparison to the previous case.

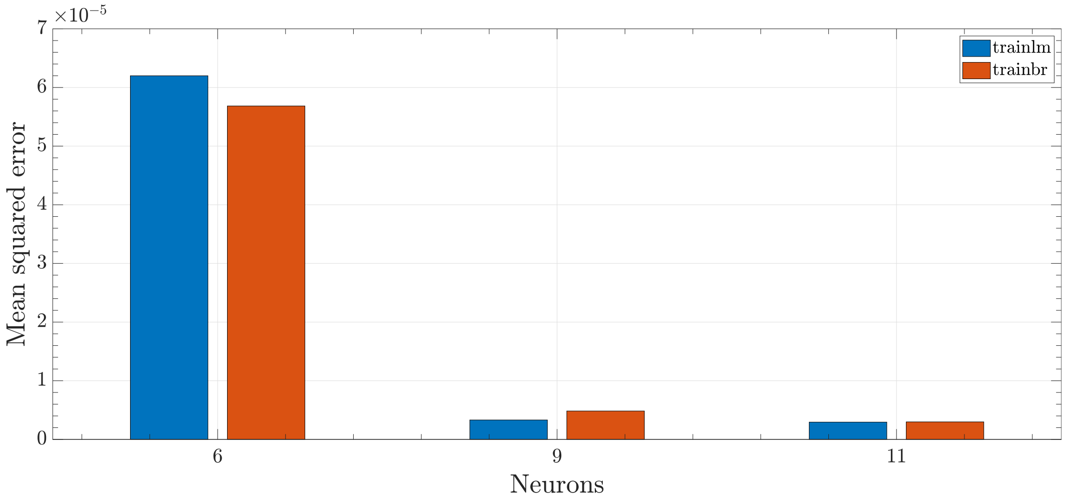


Fig. 4.8 Single hidden layer networks performance as function of neurons and training function, sampling mode 2

Differences between the two training algorithms were marginal; considering six neurons, Bayesian regression was superior, while the opposite was true when considering nine neurons; differences were basically negligible in the 11 inputs case.

Let's consider now the most complex configuration possible, with three samples per commutation, with 18 total inputs. The number of neurons was set to 8, 10, 12, 14 and 16. As predictable, increasing the neuron numbers allows the network to perform better predictions with smaller errors. In this case, the target score of 10^{-6} was almost reached by the best performers, as visible in Figure 4.9 ([52]). The Levenberg–Marquardt algorithm performed better with fewer neurons, while Bayesian regularization showed improvements in higher complexity networks.

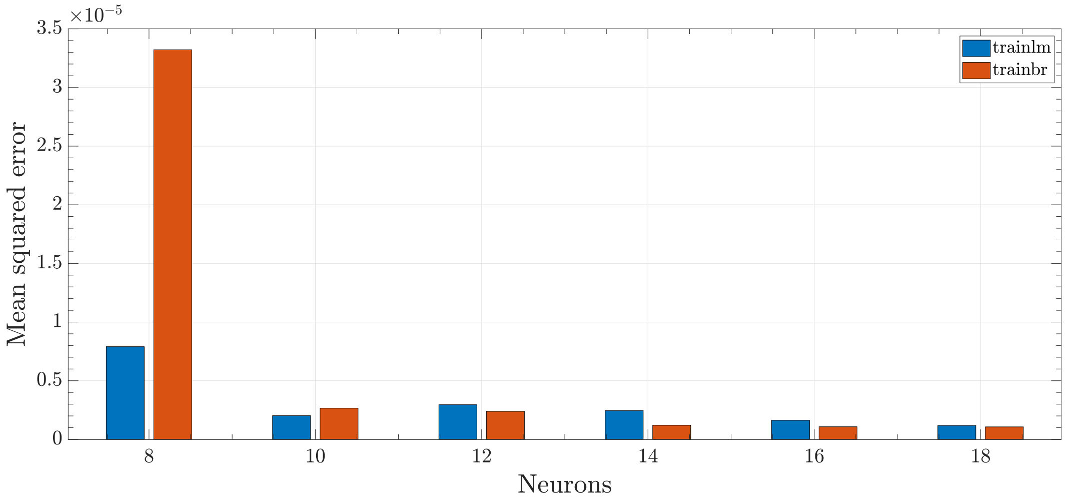


Fig. 4.9 Single hidden layer networks performance as function of neurons and training function, sampling mode 3.

In Figure 4.10 ([52]), correlation between targets/labels (i.e., expected outputs) and network outputs for sampling mode 3, 18 neurons and trainlm algorithm was reported. The high number of outliers in both scatter plots can be attributed to the polar representation of the eccentricity-phase, as with very small eccentricity values the phase error become very large. In other words, polar coordinates exhibits a singularity in the origin, thus phase isn't defined when amplitude was zero; additionally, if the amplitude value is small, the phase is ill-conditioned and large errors in fault detection would appear. However, the behavior is not so influential since it shows for very small amplitude values. In any case, the network coefficient of determination (R^2) was 0.9846 for the training dataset, 0.9836 for the validation dataset, 0.9841 for the testing datasets and 0.9844 for the complete dataset.

In Figure 4.11 ([52]), mean absolute error (MAE) boxplots have been reported for the case of single hidden layer, three input networks using a new custom verification set made of 10 different fault vectors. MAE reduction can be observed as the number of neurons increased, even though dispersion did not seem to reduce, especially for the phase fault (Figure 4.11c). The reason is the same explained just above.

Furthermore, the two training algorithms did not seem to have a marked effect on performance. Mean network errors were in the order of 10^{-3}, thus generally granting relatively accurate predictions.

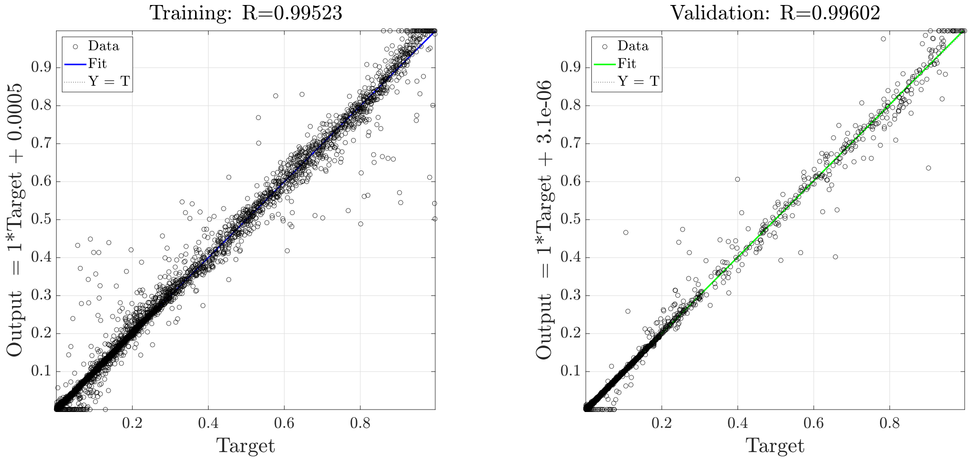
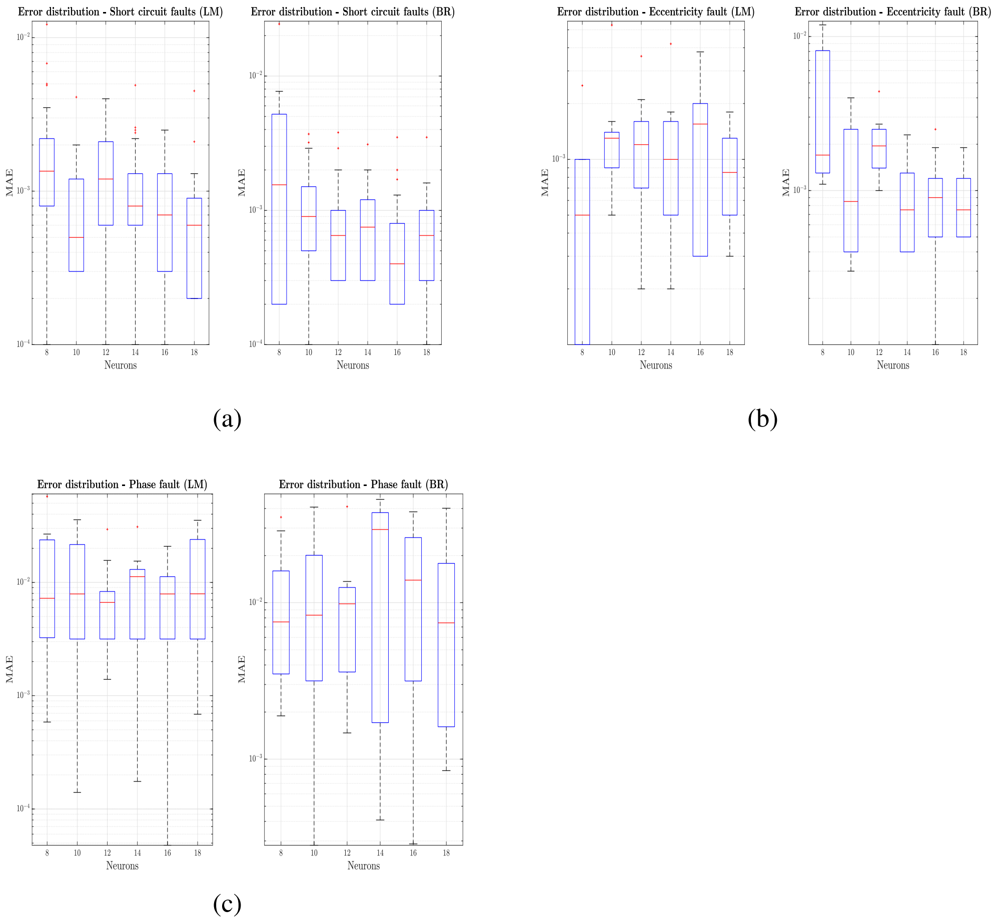


Fig. 4.10 Correlation between targets (i.e., expected outputs) and actual outputs of the network for sampling mode 3, 18 neurons and trainlm algorithm, for training and validation datasets

The last configuration tested was based on a 2-layers perceptron. Scores were not always compared to the previous topologies; this implies that an increase in the number of layers was not always associated with an increase in network performance. Additionally, the increase in network depth implied a large amount of new parameters to be evaluated, so the total training time was about three time as long as in the single-layer network.

The only configuration capable of achieving a score better than the shallow configuration was the 16-8 network using Bayesian regularization algorithm for training, as visible in Figure 4.12 ([52]). In this case, the performance score was less than the set 10^{-6}, thus achieving the best score of all the considered configurations.



(a)

(b)

(c)

Fig. 4.11 Distribution of the mean absolute error (MAE) on the identification of the considered fault modes, for increasing number of neurons in the layer. Errors were determined for a custom identification set, randomly sampled in the acceptable range. (**a**) Short circuits. (**b**) Eccentricity. (**c**) Phase.

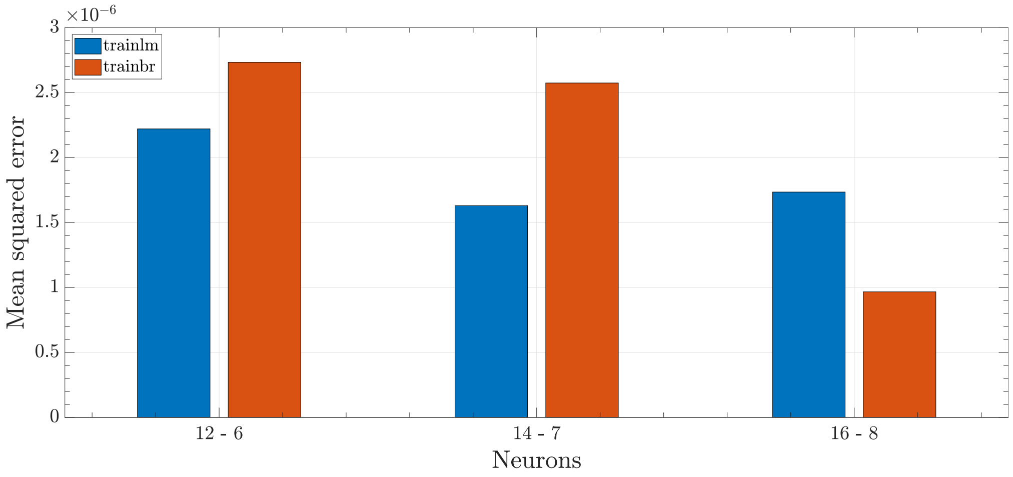


Fig. 4.12 Double hidden layer networks performance as function of neurons and training function, sampling mode 3.

4.3 Improved motor FDI

While the previous FDI algorithm achieves good accuracy, some important approximations have been assumed to make the computation easier. Chiefly among them, is the effect of the temperature on altering phase resistance. This improved algorithm allows the prediction of temperature too and also considers the phase resistance change caused by temperature variations. This improvements have been firstly presented in [58].

The algorithm can be summarized as such:

1. Faults vectors are generated, and the system is simulated using these values, obtaining a simulations dataset;
2. Relevant physical quantities are logged for each simulation (e.g., voltages, currents, motor angular position and speed);
3. In each simulation, for each phase, an estimation of the back-EMF coefficient is calculated;
4. Estimation error is minimized by obtaining the real values of phase resistance and back-EMF coefficient;
5. These values are used in a neural network to predict the health status of the system.

The first two steps are self-explanatory; the faults are modeled as in the previous section, using a fault vector $f = [N_a,\ N_b,\ N_c,\ \xi_x,\ \xi_y,\ \Delta T]$, with 6 different components: N_a, N_b, N_c which are the fraction of turns shorted for each phase; ξ_x, ξ_y which are the components of static eccentricity now in cartesian coordinates (polar coordinates were previously used and were problematic for small amplitudes) and finally ΔT which is the temperature deviation from reference conditions $T_0 = 20$ °C. Variable ranges are the ones proposed in previous section. For stator windings temperature, the range $\Delta T = [-50;\ +70]$ °C is chosen; it is representative of conditions encountered in aeronautical applications. However, the range chosen might be too restrictive to include sudden transitory temperature spikes and might need adjustments in future developments but the algorithm applies without modifications to modified fault and/or temperature ranges. The faults are injected into the model prior to each simulation.

4.3.1 Health index

Once again, the back-EMF coefficient has been used as an health index for the FDI algorithm. However, some important modifications are applied to the previous simplified approach as now the effect of the temperature is considered.

In order to do so, the temperature dependency of each phase resistance has been included in the model using the usual equation:

$$R = R_{ref}(1 + \alpha \Delta T) \tag{4.9}$$

where $\alpha = 0.00404\ ^{\circ}C^{-1}$ is copper resistance temperature coefficient, R_{ref} is the reference temperature (for this case $T_{ref} = 20\ ^{\circ}C$) and ΔT is the temperature difference from reference conditions.

The main assumption used in this approach is that the motor is purely ohmic, i.e. the inductive component is not considered. This is a strong assumptions but the algorithm is much simpler in this case and still remains accurate, so this approach will be discussed. As previously stated, the whole derivation which will be here described has been presented in [58].

The back-EMF coefficient is obtained using the following equation:

$$V - k'\omega = V - (k + k_e)\omega = R_0 i \tag{4.10}$$

where k' is the estimated back-EMF coefficient and k_e is the estimation error. In this case, values of V, ω, i are those that can be measured from the simulation (and by extension, from a real system), while R_0 is the nominal resistance of a phase. There is an error in the estimation of the back-EMF coefficient since the nominal resistance is used, thus not considering the effect of both a temperature variation and partial phase to phase short, which changes this value.

The actual (or true) system condition is described by:

$$V - k\omega = R \cdot i = (R_0 + \Delta R)i \tag{4.11}$$

where k is the true back-EMF coefficient, $R = (R_0 + \Delta R)$ is the true resistance, i.e., the effective resistance of the coil in the instantaneous conditions of temperature, fault, etc. and ΔR is the deviation of actual resistance from nominal value.

Now, subtracting Equation (4.11) from Equation (4.10) and rearranging, one can obtain:

$$k_e = \Delta R \frac{i}{\omega} \tag{4.12}$$

Assuming that ΔR is constant (i.e., R varies slowly), which is reasonable in the framework presented, since each measurement is very short (in the order of one second), applying derivative to Equation (4.12) we have:

$$\frac{\partial k_e}{\partial (i/\omega)} = \Delta R \tag{4.13}$$

Furthermore, assuming that k is constant, which implies that the fault does not change during the simulation, we can obtain:

$$\frac{\partial (k + k_e)}{\partial (i/\omega)} = \frac{\partial k'}{\partial (i/\omega)} = \Delta R \tag{4.14}$$

So, we have demonstrated that $k' = (k + k_e)$ is linearly dependent to i/ω with a slope equal to ΔR. Now, to obtain the real values of back-EMF coefficient and resistance, we have to iteratively reconstruct the value of k (Equation (4.11)), using a temporary R^* variable to optimize to make Equation (4.14) equal to zero. This stems from the definition of back-EMF:

$$|BEMF| = \frac{\partial \Phi}{\partial t} = \frac{\partial \Phi}{\partial \theta}\frac{\partial \theta}{\partial t} = \frac{\partial \Phi}{\partial \theta}\omega \quad \rightarrow \quad k_j(n_j, \theta, T) = \frac{\partial \Phi(n_j, \theta, T)}{\partial \theta} \tag{4.15}$$

where Φ is the concatenated magnetic flux, n is the number of shorted turns and T is the temperature. In other words, the concatenated magnetic flux is a function of angle, number of turns and temperature and so is the back-EMF coefficient k_j. It does depend only on motor geometry and on the magnetic properties of the magnets, and thus the temperature dependency. In this preliminary work, the temperature-induced variation of magnetic properties is not considered and will be added in further developments.

The following problem must then be solved:

$$\begin{cases} V - k^{*}\omega = R^{*} \cdot i \\ \dfrac{\partial(k^{*})}{\partial(i/\omega)} = 0 \end{cases} \tag{4.16}$$

At convergence, we obtain $R^{*} = R$ which implies $k_e = 0$ (from Equation (4.11)): we are calculating true resistance and true back-EMF coefficient. In this work, a simple bisection method is used to perform the optimization.

Up to this point, we have considered $k_j(\theta,\ i/\omega)$, as visible in Figure 4.13, where the subscript j indicates one of the three phases; in this case, the number of variables to optimize will be $3 \cdot n$, where n is the number of subdivisions along the θ axis. The total number of variables will thus be $6 \cdot n$, i.e., $3n$ values of resistance and $3n$ values of back-EMF coefficient.

However, in order to simplify the computation, the dependence on θ has been dropped, thus collapsing the 3D graph into a 2D plot of $k_j(i/\omega)$, i.e., Figure 4.14 ([58]). Now, a 'global' (or *generalized*) k_j approximation can be calculated, using least square fit. Values close to zero have been discarded, since they provide no additional information, and an absolute value on k_j has been applied. The final result is a reduction of the number of variables from $6 \cdot n$ to 6, i.e., 3 generalized resistances (one for each phase) and 3 generalized back-EMF coefficients.

These 6 values are used in a simple feed-forward neural network to perform an estimation of the fault vector f.

Figures 4.13 ([58]) and 4.14 have been obtained using a parabolic position command (i.e., a speed ramp) with a constant angular acceleration of 0.3 rad/s^2; initial conditions are zero angular position and zero angular velocity ($\theta(0) = \dot{\theta}(0) = 0$). The following fault vector was seeded: $f = [0.0375,\ 0.0504,\ 0.0507,\ 0.0059,\ 2.8 \cdot 10^{-5},\ 4]$.

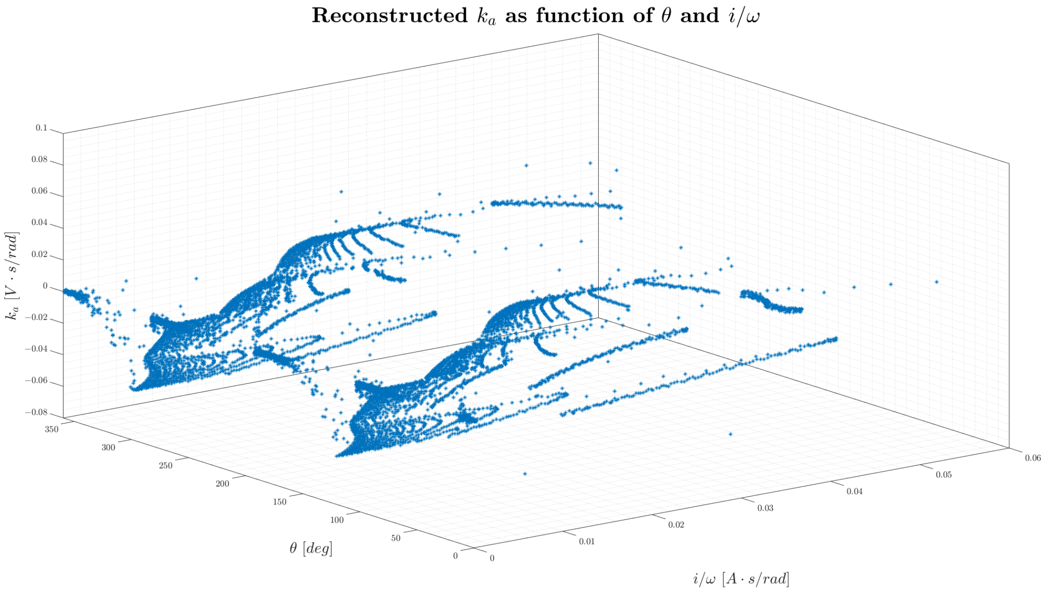


Fig. 4.13 Reconstructed k_a as function of both θ and i/ω.

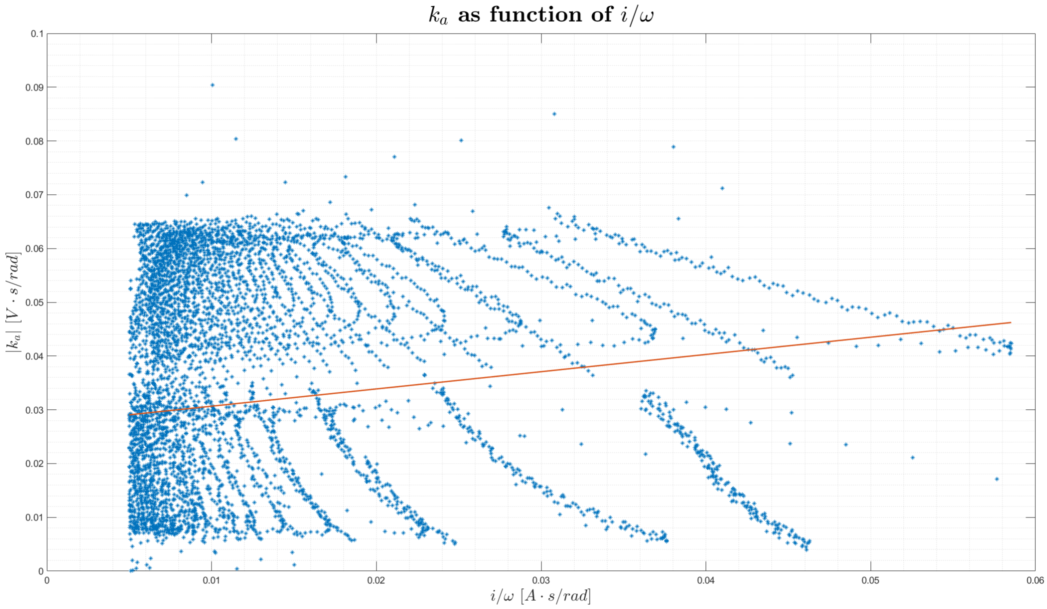


Fig. 4.14 Reconstructed k_a as function of i/ω (non optimized).

4.3.2 Dataset

The number of different fault vectors simulated is 600; dataset has been randomly divided into $70\%, 15\%, 15\%$ subsets for training, validation and testing, respectively.

Dataset size has been empirically chosen to be a good representation of the 6-dimensional fault space; the dataset, regarding the first 5 variables of the fault vector (N_a, N_b, N_c, ξ_x, ξ_y) is the same presented in the previous section, with the two eccentricity components now transformed into cartesian coordinates from polar. For each fault vector, a temperature difference value (ΔT) has been appended, being randomly sampled from the allowable interval.

Each simulation takes about 50 seconds to run, on a 6-core AMD-5600X using 3200 MHz DDR4 RAM; to speed up dataset generation, parallel pooling has been used, and also Simulink Accelerator mode provided benefits in computation time reduction.

In the future, we hope to implement the faults considered in this study on a real test bench, even though it is a complex task, especially for the partial phase shorts.

4.3.3 Results

Several tests on different hyperparameter combinations have been carried out, and the best performing set has been used for error distribution calculation. The network architecture is very similar to that presented in the previous section, with minor modifications to the hyperparameters values.

The architecture chosen is a feed-forward neural network with a single hidden layer of size 12; training function is *trainlm* which implements the Levenberg–Marquardt algorithm; activation function for each neuron is the hyperbolic tangent sigmoid. Maximum size of failed validation checks is set to 10.

As expected, the network inputs vector is of size 6, including 3 generalized back-EMF coefficients and 3 generalized resistances, while the output is again of size 6 and is the fault vector used to generate the simulation.

In Figure 4.15 ([58]), the mean absolute error box plot for each variable is shown. A new subset of 10 simulations is used and called external validation set. It is a good representation of the predictive capabilities of the network with new data during operations, since it was never used by the network during training. As visible, the mean absolute error is very small, in the order of 0.02 on normalized data. The error distribution is however uneven between variables and this might be caused by the relatively small dataset used in training.

In absolute terms, the mean error for the ΔT estimation, on the external verification dataset, is about 1.8 °C, which is a good result.

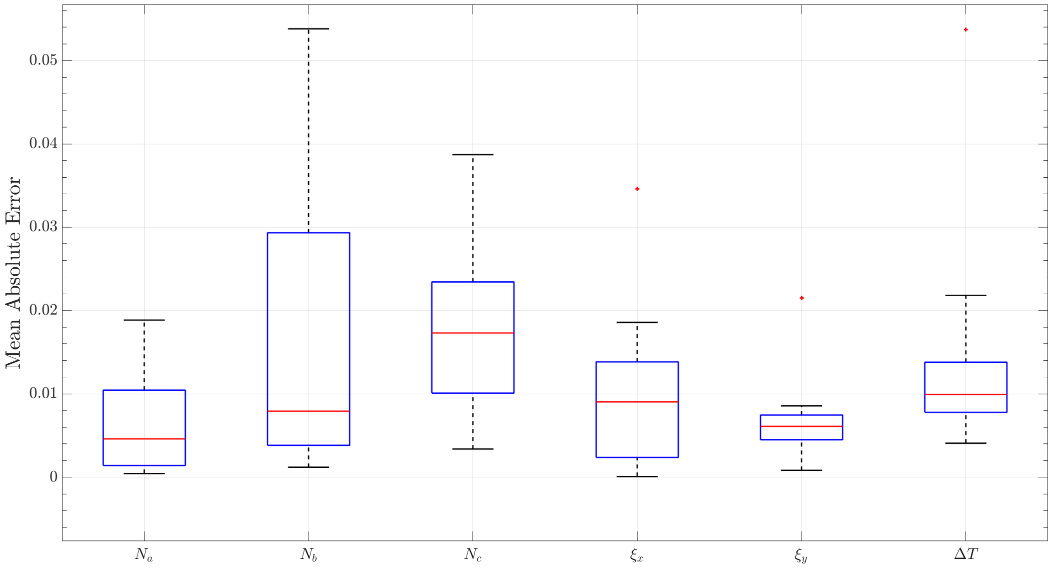


Fig. 4.15 Mean absolute error for external validation set.

4.3.4 Discussion

The strong points of the algorithms presented include the ability to estimate the current health status of the motor in terms of fault variables including partial phase shorts, static eccentricity and temperature deviation from ambient conditions. Additionally, no additional sensors besides those needed for normal operations are required, since inclusion of additional sensors it is usually expensive and a cause of lower reliability. Fault estimation is performed by a feed-forward neural network after the raw data have been pre-processed.

As with any other work that includes neural networks in the pipeline, a parametric optimization of the network could yield additional benefits in the form of higher accuracy or a simpler network if the algorithm is to be implemented on embedded hardware.

Furthermore, several assumptions have been made to simplify the algorithm, mainly the temperature independence of magnetic properties. Even though it is a reasonable assumption for small variations of temperature (magnetic flux variation of ca. -4% for 100 °C for SaCo magnets), a generalized algorithm should include and simulate such variations.

Increasing the available data, i.e. increasing the number of simulations, could prove beneficial in obtaining better prediction performance. The best combination of actuation command-load should be evaluated, which could again cover more feature space with the same length of simulation.

Finally, an empirical validation on properly calibrated equipment is mandatory to test that the assumptions made are reasonable and can effectively represent the system status, even tough it is quite hard to obtain identical motors and apply specific and precise fault conditions to them.

4.4 FDI for mechanical transmissions

Another application of the proposed FDI method has been to mechanical transmissions, as reported in [51]. The algorithm proposed is analogous to the one used for electrical faults, described in the previous section; however, this time, the health index used is the residual torque, defined as the difference between motor torque, hinge moment and inertial torque, which carries information regarding dry friction in the whole transmission. This approach is not component-based, rather a global estimate of the dry friction expressed by the whole transmission.

4.4.1 Health index

As previously stated, the health index used for this work is the residual torque, defined as:

$$T_R = T_M - T_{in} - H_e = k_{bemf} i - J_m \ddot{\theta}_m - H_e \tag{4.17}$$

where T_M and T_{in} are respectively the motor and inertial torque, k_{bemf} is the equivalent single-phase back-emf coefficient, i is the single-phase equivalent motor current, J_m is the transmission equivalent inertia (fast-shaft reduced), $\ddot{\theta}_m$ is the angular acceleration of the fast shaft and H_e is the hinge moment.

Most of these quantities are already available, wither because are needed to maintain correct motor operations (i.e. current, back-emf coefficient) or because are physical constants, such as equivalent transmission inertia. Additionally, some are easily derivable such as the angular acceleration which can be obtained by differentiating the position signal obtained by the Hall sensors, used to close the position loop.

The only quantity which is not readily available is the hinge moment, which can either be measured directly (by using a load cell) or indirectly, e.g. using a mathematical model of the aircraft-environment interaction.

In the proposed algorithm, the hinge moment is supposed to be known, thus the previous equation can be solved for each time-step to obtain the residual torque.

The fault vector in this case does not have a direct physical correlation to a specific component, but rather is a collective representation of the health status of the whole transmission. The fault vector used is:

$$f = [\eta_a, \eta_o, FSJ, FDJ] \tag{4.18}$$

where η_a and η_o are mechanical efficiencies under aiding and opposing loads, FSJ is the static friction force and FDJ is the dynamic dry friction force.

The model of dry friction implemented in the model, as explained in chapter 3, is the Borello friction model, which is a numerical-friendly implementation of the Coulumb dry friction model. Using this model, friction torque T_F can be tought as the the sum of two components - load invariant and load-dependent friction torques. In case of sliding conditions, the friction torque is equal to FDJ, otherwise it is equal to the motor torque for values lower than FSJ.

In formulae it can be expressed as:

$$T_F = \begin{cases} T_M, & \dot{x} = 0 \wedge |FM| \leq FSJ \\ FDJ \cdot sign(FM), & \dot{x} = 0 \wedge |FM| > FSJ \\ FDJ \cdot sign(\dot{x}), & \dot{x} \neq 0 \end{cases} \tag{4.19}$$

Additionally, the load-dependent friction torque component can be determined once the external load F_R is known, and is mediated by the two transmission efficiencies, as demonstrated in [59]. In other words, the total friction torque can be expressed as:

$$T_F = \begin{cases} FDJ + (1 - \eta_a) \cdot |F_R|, & \text{under Aiding loads} \\ FDJ + (\frac{1}{\eta_o} - 1) \cdot |F_R|, & \text{under Opposing loads} \end{cases} \tag{4.20}$$

Finally, thanks to the work presented in [60], an additional equation between the two efficiencies and the gear ratio τ can be written:

$$\eta_a = \frac{2\tau^2\eta_o - \tau^2 + \tau}{\tau^2\eta_o + \tau - \eta_o + 1} \tag{4.21}$$

Thus, this additional equation implicitly reduces the dimensionality of the problem, making it easier to solve. In future works, these relations can be directly used in tools such as physics-informed neural networks (PINN) to obtain better estimation accuracy or to reduce the dataset size needed to train the model.

4.4.2 Dataset

In a similar fashion to previous cases, a dataset has been generated using a numerical model of the system, which is still an electromechanical actuator acting on a primary flight control (elevator). In particular, the model used is the BLDC LF model, the same one presented in Chapter 3.

The total number of simulations is 2000, with each one having a different fault vector injected, The command curve used in this simulation has been chosen 'fill' the TR/$\dot{\theta}_m$ space in order to increase the information density of each simulations allowing to reduce the total simulation time. An example of this response is shown in Figure 4.16 (from [51]).

In particular, a third-order polynomial has been chosen with the following points as guide: (0,0) - (0.16, -15) - (0.3, 0) - (0.5, 15), in terms of (time, angular position in degrees). A representation of the command curve is shown in Fig. 4.17 (from [51]).

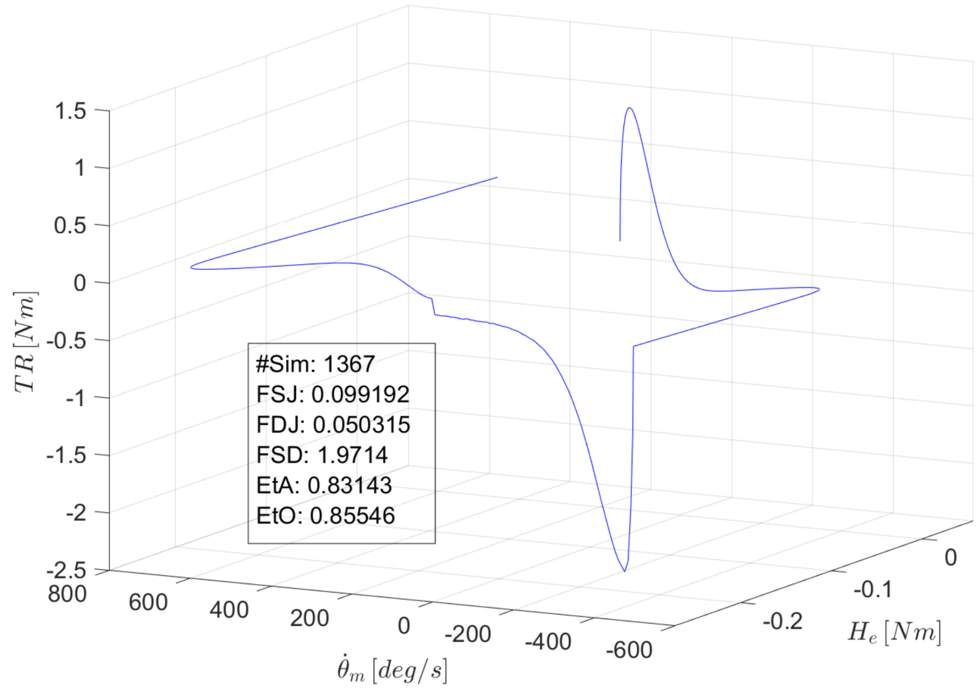


Fig. 4.16 Example of system response in terms of $\dot{\theta}_m, H_e, TR$

All the simulations use ode1 as solver (simple forward Euler integrator) with a timestep of 1 μs, and are 0.5 s long.

Two datasets have been generated: the one for case B will have the full dynamics model as explained in Chapter 3, thus relating to an actuation in flight conditions; on the other hand, case A will be performed on a model stripped of the flight dynamics component, simulating a ground actuation. Thus, for case A, the hinge moment is a constant simulating the weight of the elevator itself.

4.4.3 Artificial neural network

In a similar fashion to previous FDI algorithms, artificial neural networks are also used in this regression task to estimate the targets, i.e. the fault vector $[\eta_a, \eta_o, FSJ, FDJ]$ starting from samples of the residual torque which has been reconstructed as seem before.

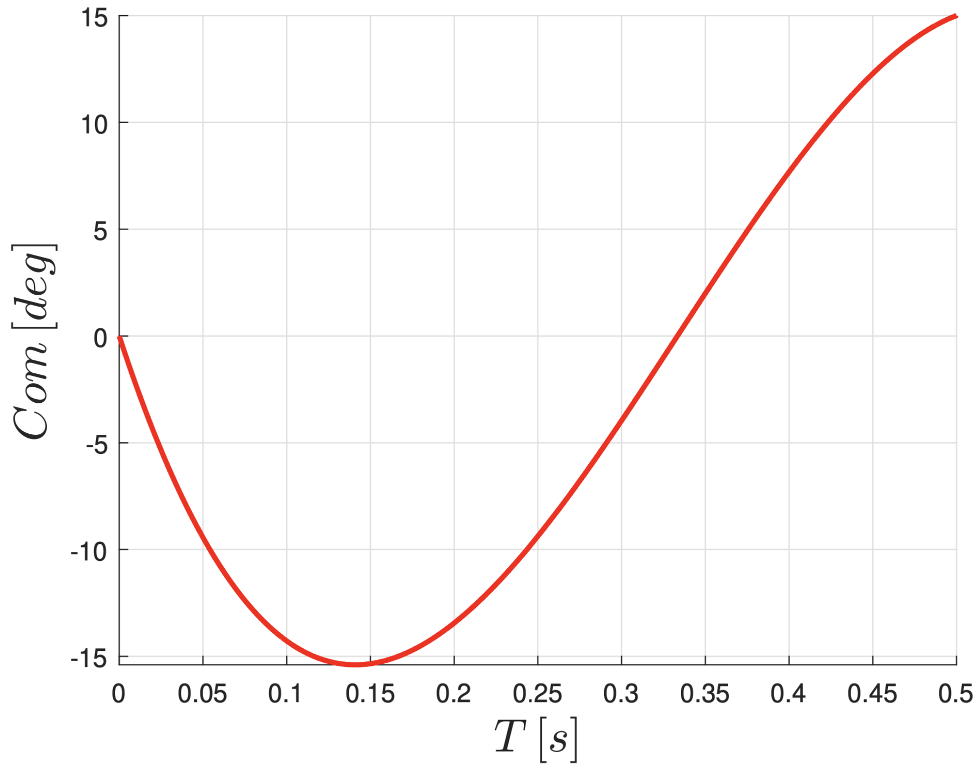


Fig. 4.17 Command curve used for the all simulations

Once again, the whole pipeline has been implemented in MATLAB using the Machine Learning toolbox, given the ease of integration between that and the data generated from the Simulink simulations.

The first task has been to reduce the size of the residual torque vector, otherwise one would have 500.000 points per each simulations - it was chosen to reduce the number of points to 2000, which seemed enough to capture most information while not having input vector with exceedingly large size. Thus, the apparent timestep is of 25 ms.

Hidden layer size has been chosen to 200 as it is a good medium value between the input size (2000) and output size. The overall architecture is shown in Fig. 4.18, however the output layer should have three element rather than 2 as depicted.

In fact, it was chosen to only estimate three elements of the fault vector, namely η_a, FSJ and FDJ, given the relation between η_a and η_o shown in Eq. 4.21. This allows a simpler network and leverages system knowledge, considering that the gear ratio τ can be considered constant during the span of each simulation.

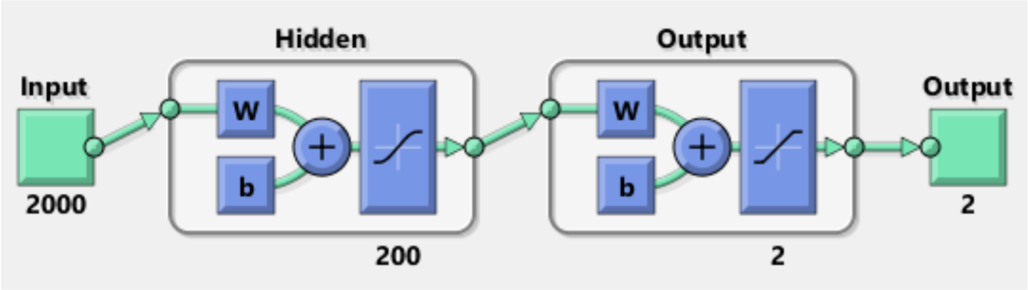


Fig. 4.18 Neural network architecture

All the data are being normalized in the range (0, 1) before being used as training vectors in the neural network, as this improves the accuracy of the prediction. The activation function used in every neuron is '*satlins*', i.e. a linear, symmetric saturated linear function which can provide non-linearity to the network and is easier to compute compared to hyperbolic tangent or sigmoid. The training function used has been '*trainscg*' i.e. scaled conjugated gradient method. This particular function has been chosen since is a first-order method and is fast and relatively light on system memory, since the size of the problem has lead to several out of memory incidents using more standard functions (e.g. 'trainlm' or 'trainbr').

Each dataset has been randomly split with the usual percentages of 70% for training, 15% for testing and 10% for validation.

Finally, as with most regression tasks, the Mean Squared Error (MSE) has been chosen as performance metric.

4.4.4 Results

Initial testing has shown that predicting the value of FSJ (static force) is not possible with the current setup, since the number of simulation steps in sticking conditions is really slim, thus not providing enough information to the network. In principle it is possible to use different commands that allows the transmission to work more in sticking conditions thus encoding more information in the residual torque curve. So, the predicted values are FDJ (dynamic force) and η_o, and thus η_a indirectly.

Case A: ground test

As previously state, these simulations replace the dynamics of the aircraft with a simple constant block, and are meant to simulate a ground actuation where only the weight of the elevator is present.

Performance are (2.41, 2.43, 2.45)$\cdot 10^{-4}$ for the training, testing and validation subsets, respectively.

Error values are in the range a few percentage points, as shown in Table 4.1. A summarizing view is shown in Fig. 4.19, where on the x-axis the targets are shown, while on the y-axis the prediction of the networks are shown. In a perfect condition, all the predictions (red marks) are aligned along the y = x curve.

Table 4.1 10 random samples for FDJ and relative predictions (Case A)

Target	Prediction	Rel. Error [%]
0.9782	0.9792	0.11
0.8399	0.8442	0.50
0.2670	0.2722	1.98
0.2687	0.2614	2.73
0.5300	0.5160	2.64
0.6933	0.7047	1.64
0.4183	0.4505	7.69
0.6042	0.6013	0.49
0.6015	0.5843	2.85
0.8478	0.8707	2.71

Case B: generic in-flight actuation

In this more complex case, the whole longitudinal dynamic of the aircraft is considered, thus simulating a generic in-flight actuation. However, the particular combinations of command and initial flight conditions are not completely general and should be tailored to the specific aircraft.

Results obtained in this more complex situation are still satisfying, with MSE accuracy in the order of (1.742, 1.926, 2.601)$\cdot 10^{-6}$ for the best tested combination, which has an hidden layer of size 50 and uses 500 simulations as dataset.

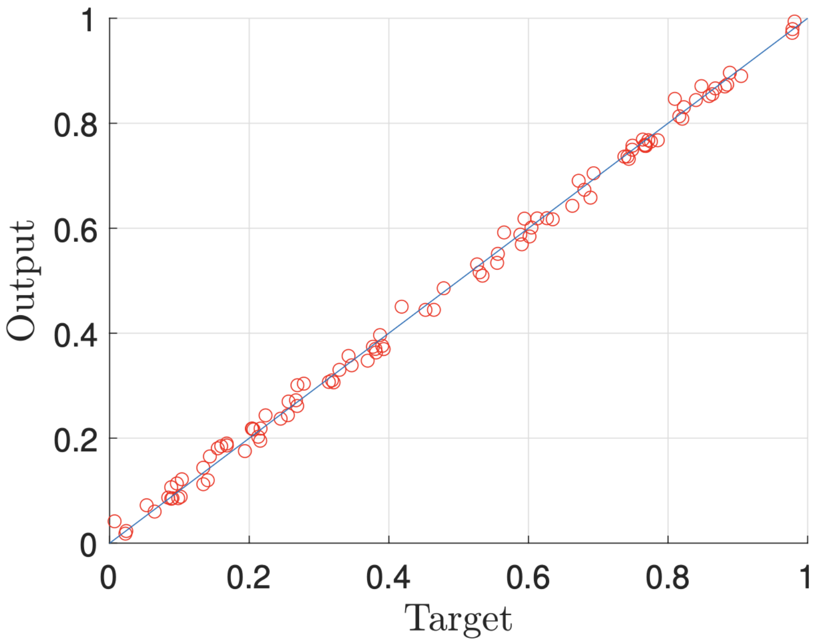


Fig. 4.19 Validation set outputs, case A

The average MSE error are much lower than those of Case A. The main explanation is that in Case A the external load is constant, and thus is equal for all timesteps. This leads to loss of information and the hinge moment does not provide more information to the network. On the other hand, in Case B there is a new variable which contributes to the learning of the network, and this drives down the average MSE. In fact, in Figures 4.20 and 4.21 the fit capabilites of the network are shown. Results are globally good and there isn't much difference between the two variables.

Several combinations of hidden layer size, inputs size and number of simulations have been tested, and results are shown in Table 4.2.

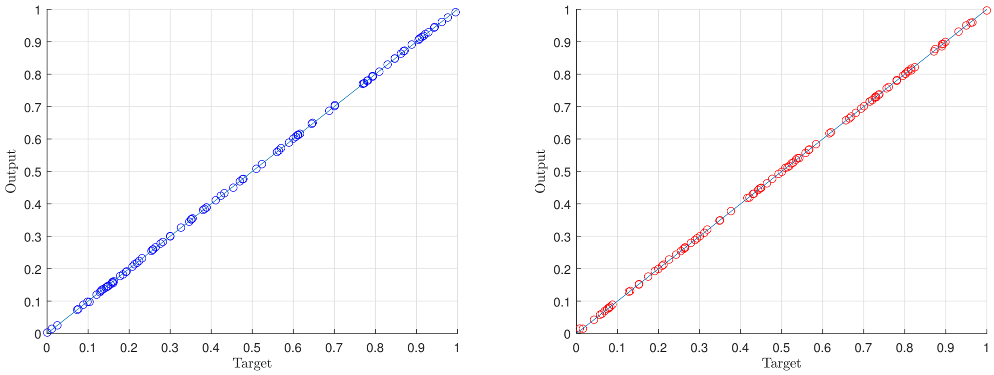
Similarly to Case A, 10 random samples and relative predictions are shown in Table 4.3.

Table 4.2 Settings tried for Case B and relative performances.

Timesteps	Simulations	Neurons	Training MSE	Test MSE
100	100	10	$6.007 \cdot 10^{-6}$	$1.611 \cdot 10^{-5}$
500	100	50	$4.364 \cdot 10^{-5}$	$4.244 \cdot 10^{-5}$
100	500	10	$6.217 \cdot 10^{-6}$	$6.730 \cdot 10^{-6}$
100	500	50	$4.529 \cdot 10^{-6}$	$7.643 \cdot 10^{-6}$
500	500	50	$1.742 \cdot 10^{-6}$	$2.601 \cdot 10^{-6}$

Table 4.3 Validation for Case B. 10 random predicted values were confronted with their actual value (“Target value”) to calculate the relative error of the prediction.

Parameter: FDJ			**Parameter:** η_O		
Target Value	**Predicted Value**	**Relative Error [%]**	**Target Value**	**Predicted Value**	**Relative Error [%]**
0.9648	0.9596	0.54	0.0761	0.0756	0.73
0.5432	0.5415	0.30	0.0734	0.0730	0.45
0.3115	0.3107	0.26	0.6876	0.6871	0.08
0.4217	0.4195	0.51	0.0984	0.0979	0.57
0.5003	0.4987	0.32	0.5235	0.5224	0.22
0.5839	0.5842	0.06	0.6001	0.6012	0.19
0.2609	0.2608	0.02	0.8887	0.8914	0.31
0.2121	0.2123	0.12	0.7008	0.7015	0.10
0.9311	0.9312	0.02	0.1336	0.1349	0.96
0.7147	0.7158	0.16	0.6471	0.6499	0.43



(a) **(b)**

Fig. 4.22 The values on the X axes are the actual values of the parameters (normalized between 0 and 1), while the values on the Y axes are those predicted by the network. This figure shows the individual fits for: **(a)** FDJ of Case B; **(b)** η_O of Case B

Chapter 5

Conclusions

The objective of this thesis was to present several tools that are useful to improve the smartness of electromechnical actuators, especially for aerospace use, which implies that the system is capable of detecting faults autonomously.

In this sense, several fault detection and identification algorithms have been presented, with capabilities ranging from mechanical faults, to partial electric shorts to static eccentricity. All of this models leverages machine learning tools and have been trained using data provided by numerical models of a real system. Preliminary results are good but as always validation on data from real systems is mandatory before deployment on active systems.

On this topic, different MATLAB-Simulink numerical models, with varying levels of fidelity, have been developed for this task. High fidelity models can be used to generate realistic, synthetic data when either data are not available or it is too complex or hard to obtain large amount of data relative to a particular fault with a specific magnitude. The main advantage of this approach is the accuracy of the fault modeling which can be tailored to specific values. Albeit some testing has been performed on validating the model on a real system, additional testing is needed to evaluate the accuracy of the fault modeling included in the models, and to supplement the implementation proposed.

Finally, several FBG-based optical sensors have been proposed, for vibrations, temperature and mechanical strain. These preliminary designs will need further testing to optimize the design in terms of accuracy and robustness. The final objective is to integrate measure obtained by said sensors in a more complete FDI scheme, and possibly further in a prognostics tool that is capable of also evaluating the remaining useful life of components or of the whole system.

References

[1] Prashanthi Boddu and Liudong Xing. Reliability evaluation and optimization of series–parallel systems with k-out-of-n: G subsystems and mixed redundancy types. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, 227(2):187–198, 2013.

[2] George J Vachtsevanos, Frank Lewis, Michael Roemer, Andrew Hess, and Biqing Wu. *Intelligent fault diagnosis and prognosis for engineering systems*, volume 456. Wiley Hoboken, 2006.

[3] Farzaneh Ahmadzadeh and Jan Lundberg. Remaining useful life estimation. *International Journal of System Assurance Engineering and Management*, 5:461–474, 2014.

[4] Rosmaini Ahmad and Shahrul Kamaruddin. An overview of time-based and condition-based maintenance in industrial application. *Computers & industrial engineering*, 63(1):135–149, 2012.

[5] Mihaela Mitici and Ingeborg De Pater. Online model-based remaining-useful-life prognostics for aircraft cooling units using time-warping degradation clustering. *Aerospace*, 8(6):168, 2021.

[6] Changchang Che, Huawei Wang, Qiang Fu, and Xiaomei Ni. Combining multiple deep learning algorithms for prognostic and health management of aircraft. *Aerospace Science and Technology*, 94:105423, 2019.

[7] Muhammad Arif Riza, Yun Ii Go, Robert RJ Maier, Sulaiman Wadi Harun, and Siti Barirah Ahmad Anas. Development of fbg humidity sensor via controlled annealing temperature of additive enhanced zno nanostructure coating. *Optical Fiber Technology*, 68:102802, 2022.

[8] Jianxin Zhang, Xueyun Shen, Miao Qian, Zhong Xiang, and Xudong Hu. An optical fiber sensor based on polyimide coated fiber bragg grating for measurement of relative humidity. *Optical Fiber Technology*, 61:102406, 2021.

[9] Jitendra Kumar, Gitender Singh, Manoj K Saxena, Om Prakash, Sudhir K Dixit, and Shankar V Nakhe. Development and studies on fbg temperature sensor for applications in nuclear fuel cycle facilities. *IEEE Sensors Journal*, 21(6):7613–7619, 2020.

[10] Guang Yan, Tianqi Wang, Lianqing Zhu, Fanyong Meng, and Wei Zhuang. A novel strain-decoupled sensitized fbg temperature sensor and its applications to aircraft thermal management. *Optics & Laser Technology*, 140:106597, 2021.

[11] Flavio Esposito, Stefania Campopiano, and Agostino Iadicicco. Miniaturized strain-free fiber bragg grating temperature sensors. *IEEE Sensors Journal*, 22(17):16898–16903, 2022.

[12] Ole Krarup, Chams Baker, Liang Chen, and Xiaoyi Bao. Nonlinear resolution enhancement of an fbg based temperature sensor using the kerr effect. *Optics Express*, 28(26):39181–39188, 2020.

[13] Gautam Hegde, MVN Prasad, and S Asokan. Temperature compensated diaphragm based fiber bragg grating (fbg) sensor for high pressure measurement for space applications. *Microelectronic Engineering*, 248:111615, 2021.

[14] Qiming Shu, Lei Wu, Shunzhi Lu, and Wensheng Xiao. High-sensitivity structure based on fiber bragg grating sensor and its application in nonintrusive detection of pipeline pressure change. *Measurement*, 189:110444, 2022.

[15] Shiuh-Chuan Her and Shin-Zhong Weng. Fiber bragg grating pressure sensor integrated with epoxy diaphragm. *Sensors*, 21(9):3199, 2021.

[16] BS Kavitha, S Sridevi, Pandeeswar Makam, Debasis Ghosh, Thimmaiah Govindaraju, S Asokan, and AK Sood. Highly sensitive and rapid detection of mercury in water using functionalized etched fiber bragg grating sensors. *Sensors and Actuators B: Chemical*, 333:129550, 2021.

[17] AS Lalasangi, JF Akki, KG Manohar, T Srinivas, P Radhakrishnan, Sanjay Kher, NS Mehla, and US Raikar. Fiber bragg grating sensor for detection of nitrate concentration in water. *Sensors & Transducers*, 125(2):187, 2011.

[18] Kerstin Schroeder, Wolfgang Ecke, and Reinhardt Willsch. Optical fiber bragg grating hydrogen sensor based on evanescent-field interaction with palladium thin-film transducer. *Optics and Lasers in Engineering*, 47(10):1018–1022, 2009.

[19] AA Jørgensen, D Kong, MR Henriksen, F Klejs, Z Ye, OB Helgason, HE Hansen, H Hu, M Yankov, S Forchhammer, et al. Petabit-per-second data transmission using a chip-scale microcomb ring resonator source. *Nature Photonics*, pages 1–5, 2022.

[20] Alexwind. Optical fiber transmission windows.svg, 2009. Retrieved on 07-03-2023. URL: https://commons.wikimedia.org/wiki/File:Optical_fiber_transmission_windows.svg.

[21] Zátonyi Sándor. Fénytörés.jpg, 2005. Retrieved on 07-03-2023. URL: https://commons.wikimedia.org/wiki/File:F%C3%A9nyt%C3%B6r%C3%A9s.jpg.

[22] Mrzeon. Différents types de fibres optiques : fibre multimode à saut d'indice, fibre multimode à gradient d'indice, fibre monomode, 2007. Retrieved on 07-03-2023. URL: https://it.m.wikipedia.org/wiki/File:Optical_fiber_types.svg.

[23] Irina Severin, Rochdi El Abdi, and Marcel Poulain. Strength measurements of silica optical fibers under severe environment. *Optics & Laser Technology*, 39(2):435–441, 2007.

[24] CR Elliott and GR Newns. Near infrared absorption spectra of silica: Oh overtones. *Applied Spectroscopy*, 25(3):378–379, 1971.

[25] John M Senior and M Yousif Jamro. *Optical fiber communications: principles and practice*. Pearson Education, 2009.

[26] CR Kurkjian. Mechanical properties of phosphate glasses. *Journal of Non-Crystalline Solids*, 263:207–212, 2000.

[27] CM Okonkwo, E Tangdiongga, H Yang, D Visani, S Loquai, R Kruglov, B Charbonnier, M Ouzzif, I Greiss, O Ziemann, et al. Recent results from the eu pof-plus project: Multi-gigabit transmission over 1 mm core diameter plastic optical fibers. *Journal of lightwave technology*, 29(2):186–193, 2011.

[28] User-A1. Modified chemical vapour deposition fibre process, 2009. Retrieved on 07-03-2023. URL: https://commons.wikimedia.org/wiki/File:OF-MCVD.svg.

[29] Jing Jin, Jiliang He, Ningfang Song, Kun Ma, and Linghai Kong. A compact four-axis interferometric fiber optic gyroscope based on multiplexing for space application. *Journal of Lightwave Technology*, 38(23):6655–6663, 2020.

[30] Steven Trpkovski, Scott A Wade, Gregory W Baxter, and Stephen F Collins. Dual temperature and strain sensor using a combined fiber bragg grating and fluorescence intensity ratio technique in er 3+-doped fiber. *Review of scientific instruments*, 74(5):2880–2885, 2003.

[31] Doğangün Uzun, Okan Ülgen, and Özgür Kocatürk. Optical force sensor with enhanced resolution for mri guided biopsy. *IEEE Sensors Journal*, 20(16):9202–9208, 2020.

[32] Shihai Zhang, Bin Liu, and Jianping He. Pipeline deformation monitoring using distributed fiber optical sensor. *Measurement*, 133:208–213, 2019.

[33] Marcelo A. Soto, Xabier Angulo-Vinuesa, Sonia Martin-Lopez, Sang-Hoon Chin, Juan Diego Ania-Castañon, Pedro Corredera, Etienne Rochat, Miguel Gonzalez-Herraez, and Luc Thévenaz. Extending the real remoteness of long-range brillouin optical time-domain fiber analyzers. *Journal of Lightwave Technology*, 32(1):152–162, 2014.

[34] Sakurambo. Fiber bragg grating-en.svg, 2008. Retrieved on 17-03-2023. URL: https://en.wikipedia.org/wiki/File:Fiber_Bragg_Grating-en.svg.

[35] Andreas Othonos, Kyriacos Kalli, David Pureur, and Alain Mugnier. *Fibre bragg gratings*. Springer, 2006.

[36] John Canning. Fibre gratings and devices for sensors and lasers. *Laser & Photonics Reviews*, 2(4):275–289, 2008.

[37] Grahamwild. Fbg3.gif, 2007. Retrieved on 17-03-2023. URL: https://en.wikipedia.org/wiki/File:Fbg3.GIF.

[38] Grahamwild. Fbg4.gif, 2007. Retrieved on 17-03-2023. URL: https://en.wikipedia.org/wiki/File:Fbg4.GIF.

[39] Kuo Li, Tommy HT Chan, Man Hong Yau, Theanh Nguyen, David P Thambiratnam, and Hwa Yaw Tam. Very sensitive fiber bragg grating accelerometer using transverse forces with an easy over-range protection and low cross axial sensitivity. *Applied Optics*, 52(25):6401–6410, 2013.

[40] Yunshan Zhang, Weigang Zhang, Yanxin Zhang, Lei Chen, Tieyi Yan, Song Wang, Lin Yu, and Yanping Li. 2-d medium–high frequency fiber bragg gratings accelerometer. *IEEE Sensors Journal*, 17(3):614–618, 2016.

[41] Li Xiong, Yongxing Guo, Wanhuan Zhou, Min Chen, and Xinglin Zhou. Fiber bragg grating-based three-axis vibration sensor. *IEEE Sensors Journal*, 21(22):25749–25757, 2021.

[42] Li Wei, Dazhou Jiang, Lingling Yu, Hengchun Li, and Zhuang Liu. A novel miniaturized fiber bragg grating vibration sensor. *IEEE Sensors Journal*, 19(24):11932–11940, 2019.

[43] G Quattrocchi, PC Berri, MDL Vedova Dalla, and P Maggiore. Optical fibers applied to aerospace systems prognostics: design and development of new fbg-based vibration sensors. In *IOP Conference Series: Materials Science and Engineering*, volume 1024, page 012095. IOP Publishing, 2021.

[44] A Aimasso, M D L Dalla Vedova, A Esposito, and P Maggiore. Analysis of the fixing process of fbg optical sensors for thermomechanical monitoring of aerospace applications. In *12th EASN International Conference (2022)*. IOP Publishing, 2023. In press.

[45] Antonio Marotta. Development of optical sensors with fbg for vibrational analysis, 2022. *M.Sc. Thesis*, Politecnico di Torino, 2022.

[46] Shiuh-Chuan Her and Wei-Nan Lin. Simultaneous measurement of temperature and mechanical strain using a fiber bragg grating sensor. *Sensors*, 20(15):4223, 2020.

[47] William T. Moye. Eniac: The army-sponsored revolution, 1996. Retrieved on 05-05-2023. URL: https://web.archive.org/web/20170521072638/http://ftp.arl.mil/~mike/comphist/96summary/index.html.

[48] Pier C. Berri. Design and development of algorithms and technologies applied to prognostics of aerospace systems. *PhD Thesis*, Politecnico di Torino, 2021.

[49] Lorenzo Borello and Matteo DL Dalla Vedova. Dry friction discontinuous computational algorithms. *International Journal of Engineering and Innovative Technology (IJEIT)*, 3(8):1–8, 2014.

[50] Brian Stevens and F Frank. l., lewis, f., l.: Aircraft control and simulation, 1992.

[51] Gaetano Quattrocchi, Alessandro Iacono, Pier C Berri, Matteo DL Dalla Vedova, and Paolo Maggiore. A new method for friction estimation in ema transmissions. *Actuators*, 10(8):194, 2021.

[52] Gaetano Quattrocchi, Pier C Berri, Matteo DL Dalla Vedova, and Paolo Maggiore. Innovative actuator fault identification based on back electromotive force reconstruction. In *Actuators*, volume 9, page 50. MDPI, 2020.

[53] Gaetano Quattrocchi, P Berri, M Dalla Vedova, and Paolo Maggiore. Back-emf reconstruction for electromechanical actuators in presence of faults. In *Proceedings of the 30th European Safety and Reliability Conference and the 15th Probabilistic Safety Assessment and Management Conference (ESREL2020/PSAM15), Venice, Italy*, pages 1–5, 2020.

[54] Fionn Murtagh. Multilayer perceptrons for classification and regression. *Neurocomputing*, 2(5-6):183–197, 1991.

[55] Donald W. Marquardt. An Algorithm for Least-Squares Estimation of Nonlinear Parameters. *Journal of the Society for Industrial and Applied Mathematics*, 1963.

[56] Martin T. Hagan and Mohammad B. Menhaj. Training Feedforward Networks with the Marquardt Algorithm. *IEEE Transactions on Neural Networks*, 1994.

[57] F. Dan Foresee, M. T. Hagan, F. Dan Foresee, and M. T. Hagan. Gauss-Newton approximation to bayesian learning. In *IEEE International Conference on Neural Networks - Conference Proceedings*, 1997.

[58] Gaetano Quattrocchi, Pier C Berri, Matteo DL Dalla Vedova, and Paolo Maggiore. An improved fault identification method for electromechanical actuators. *Aerospace*, 9(7):341, 2022.

[59] L Borello, MDL Dalla Vedova, et al. A dry friction model and robust computational algorithm for reversible or irreversible motion transmissions. *International Journal of Mechanics and Control*, 13(2):37–48, 2012.

[60] PC Berri, MDL Dalla Vedova, P Maggiore, A Manuello Bertetto, et al. Evaluation of aiding and opposing load efficiency for an actuator planetary drive. *International Journal of Mechanics and Control*, 21(1):3–10, 2020.

Appendix A

Code

EMA model

Data file for EMA model

```
%% Initialization
clear; clc;

%% Simulation parameters


% [s] Sim. stop time
simulation.TiBr      = 3;

% [s] Timestep
simulation.DT        = 1e-6;

% [rad]  Slow shaft initial pos.
simulation.initPos   = 0;

 % [s] Filter char. time
simulation.tauFilter = 5e-5;

```

```
% [s] Com time points
simulation.timeCom   = [0 0.1 0.10001 2];

% [deg] Com value points
simulation.com       = [0 0 0.01 0.01];

% Load time points
simulation.timeLd    = [0 simulation.TiBr];

% Load value points
simulation.ld        = [0 0];

% [s] Output downsample
simulation.out_sample = 1e-3;



%% Controller


% Com flag: - pos. = 1 ; - speed = 0
controller.Flag    = 1;

% [1/s] Controller prop. gain
controller.Gprop    = 500;

% [rad/s] Pos. error saturation
controller.W_refMax = 8000*pi/30;

% [A] I_ref saturation
controller.I_Max    = 22.5;

% [] Noise amplitude factor
controller.Knoise   = 0;

```

```
% [Nms/rad] PID Proportional gain
controller.PID.GAP  = 0.2368;

% [Nm/rad] PID integrative gain
% controller.PID.GAI  = 0;

% [Nms^2/rad]   PID derivative gain
% controller.PID.GAD  = 0;


% [Nm] Max integrative error
controller.PID.ErIM = 100;

% [rad] Integrative branch tolerance
controller.PID.band = 1e-3;

% [s] Charact. time integr. branch
controller.PID.Ti   = 0.01;

% [s] Charact. time anti-windup
controller.PID.Tt   = 0.01;


% [s] Charact. time deriv. branch
controller.PID.Td   = 0;

% [1/s] Bandwidth deriv. branch
controller.PID.N    = 1000;



%% Inverter

% [A] Hystersis band amplitude
inverter.PWM.hb           = 0.1;
```

```

% [V] Supply voltage
inverter.Hbridge.Vdc       = 380;

% [ohm] Snubber resistance
inverter.Hbridge.RSnubber = 1e5;

% [F] Snubber capacity
inverter.Hbridge.CSnubber = 1e9;

% [ohm] Ron (in Universal Bridge)
inverter.Hbridge.Ron       = 1e-2;


%% Motor

% [] Pole pairs
BLDC.P    = 4;

% [] Active turns percents (phases A, B, C)
BLDC.Nabc = [1 1 1];

% [ohm] Nominal phase-to-phase resistance (e.g. Rab)
BLDC.Rs   = 2.94660867034355;

% [H] Nominal phase-to-phase inductance (e.g. Lab)
BLDC.Ls   = 30.5e-4;

% [Nm/A] BEMF constant
BLDC.Ke   = 0.957651428671564;


% [Nm] Torque saturation
BLDC.TMM  = 11.842;

```

```

% [] Static eccentricity amplitude
BLDC.zeta = 0;

% [] Static eccentricity phase
BLDC.phi  = 0;



%% Motor-transmission dynamical model


% [] Transmission ratio
dynamics.tau = 1/124;

% [kg*m^2] Motor inertia moment
dynamics.JM = 0.00077;

% [N*m*(s/rad)^(1/2)] Motor viscous damping coefficient
dynamics.CM = 0.0379430075528975;

% [kg*m^2] User inertia moment reduced on fast shaft
dynamics.JU = 0.00203912312500147;

% [N*m*s/rad] User viscous dampinig coefficient
dynamics.CU = 0.000798437115408432;

% [] Motor dry friction (% of BLDC.TMM)
dynamics.friction.FDTm = 0.00585477490946900;

% [] User dry fruction reduced to fast shaft (% of BLDC.TMM)
dynamics.friction.FDTu = 0.00990871474413106;

dynamics.a1 = 0.006683;
dynamics.b1 = 0*1.488e-8;
dynamics.c1 = 8.728;

dynamics.a2 = 0.04842/2;
dynamics.b2 = 0*3.958e-5;
dynamics.c2 = 4.633;

% [rad] Backlash width (slow shaft)
dynamics.BLK = 1e-5;

% [rad] Lower excursion limit
dynamics.ThUmin        = -2;

% [rad] Upper excursion limit
dynamics.ThUmax        = 2;
```