UNIVERSITÀ
DEGLI STUDI
DE L'AQUILA

**Department of Information Engineering,
Computer Science and Mathematics**

Master's Degree in Computer and Systems Engineering

Modeling and Control of McKibben Artificial Muscle – Application of Genetic Algorithm and Metaheuristic Optimization

First Supervisor:
Prof. Costanzo Manes

Candidate:
Michele Iessi, 248961

Second Supervisor:
Prof. Kazuhisa Ito

Identification number:
248961

Academic Year 2017–2018

MODELING AND CONTROL OF MCKIBBEN ARTIFICIAL MUSCLE – APPLICATION OF GENETIC ALGORITHM AND METAHEURISTIC OPTIMIZATION

MICHELE IESSI

Master's Degree in Computer and Systems Engineering
Department of Information Engineering, Computer Science and Mathematics
University of L'Aquila

March 23 2019

Michele Iessi: *Modeling and Control of McKibben Artificial Muscle – Application of Genetic Algorithm and Metaheuristic Optimization,* , © March 23 2019
SUPERVISORS:
Costanzo Manes
Kazuhisa Ito

ACKNOWLEDGEMENTS

Put your acknowledgements here.

CONTENTS

I THESIS 1

1 INTRODUCTION 2
- 1.1 Background and Motivation . . . 2
- 1.2 Modeling of McKibben Artificial Muscle . . . 2
- 1.3 Aim of the Study . . . 3

2 COMPOSITION OF THE THESIS 4

3 TAP WATER-DRIVEN MCKIBBEN ARTIFICIAL MUSCLE 5

4 MODELLING OF MCKIBBEN ARTIFICIAL MUSCLES 6
- 4.1 Introduction . . . 6
- 4.2 Static Model . . . 7
  - 4.2.1 Introduction . . . 7
  - 4.2.2 Static Model of the Muscle . . . 7
  - 4.2.3 Modified Static Model . . . 8
- 4.3 Muscle Model Based on System Identification . . . 9
  - 4.3.1 Identification Experiments . . . 9
- 4.4 Hysteresis Modeling . . . 12
- 4.5 Models of Hysteresis . . . 12
- 4.6 The Bouc-Wen Model of Hysteresis . . . 13
  - 4.6.1 Classic Bouc-Wen Hysteresis Model . . . 13
  - 4.6.2 Generalized Bouc-Wen Hysteresis Model . . . 13
- 4.7 Hysteresis Variable Approximation . . . 14
  - 4.7.1 Classic Bouc-Wen Model . . . 14
  - 4.7.2 Generalized Bouc-Wen Model . . . 14
- 4.8 Identified Model Summation . . . 15
  - 4.8.1 Classic Bouc-Wen Model . . . 15
  - 4.8.2 Generalized Bouc-Wen Model . . . 15
- 4.9 Evaluation of Proposed Models . . . 15
  - 4.9.1 Classic Bouc-Wen Model Parameters . . . 16
  - 4.9.2 Generalized Bouc-Wen Model Parameters . . . 17
- 4.10 Weight Change Tolerance . . . 18

5 HYSTERESIS PARAMETER OPTIMIZATION 19
- 5.1 Introduction . . . 19
- 5.2 Evolutionary Algorithms . . . 19
  - 5.2.1 Genetic Algorithm . . . 20
    - 5.2.1.1 Algorithm Overview . . . 20
    - 5.2.1.2 Algorithm Description . . . 20
    - 5.2.1.3 Initialization Step . . . 20
    - 5.2.1.4 Evaluation Step . . . 21
    - 5.2.1.5 Crossover Step . . . 22
    - 5.2.1.6 Mutation Step . . . 23
- 5.3 Metaheuristic Algorithms . . . 23
  - 5.3.1 Firefly Algorithm . . . 24
    - 5.3.1.1 Algorithm Overview . . . 25
    - 5.3.1.2 Algorithm Description . . . 25

5.3.1.3 Light Intensity and Attractiveness . . . 25
5.3.1.4 Firefly Movement . . . 26
5.3.2 Modified Firefly Algorithm . . . 27
5.3.2.1 Attractiveness at Zero Distance . . . 27
5.3.2.2 Light Absorption Coefficient . . . 27
5.3.2.3 Randomization Parameter . . . 27
5.3.3 Particle Swarm Optimization . . . 28
5.3.3.1 Algorithm Overview . . . 29
5.3.3.2 Algorithm Description . . . 29
5.3.3.3 Velocity and Position Update . . . 29
5.3.3.4 Inertia Weight . . . 30
5.3.3.5 Acceleration Constants . . . 30
5.4 Algorithm Comparison . . . 30
5.4.1 System Specifications . . . 30
5.4.2 Solution Comparison – Classic Bouc-Wen Model . . . 30
5.4.2.1 Genetic Algorithm – Classic Bouc-Wen Model . . . 30
5.4.2.2 Modified Firefly Algorithm – Classic Bouc-Wen Model 35
5.4.2.3 Particle Swarm Optimization – Classic Bouc-Wen Model 37
5.4.3 Solution Comparison – Generalized Bouc-Wen Model . . . 40
5.4.3.1 Genetic Algorithm – Generalized Bouc-Wen Model . 40
5.4.3.2 Modified Firefly Algorithm – Generalized Bouc-Wen Model . . . 44
5.4.3.3 Particle Swarm Optimization – Generalized Bouc-Wen Model . . . 47
5.4.4 Time Comparison . . . 49
6 CONTROLLER DESIGN 50
6.1 Introduction . . . 50
6.2 PI Controller Development . . . 50
6.2.1 Introduction on PID Control . . . 50
6.2.2 Application of PI Control . . . 51
6.2.2.1 Application on Muscle – 4.5 Kilograms Weight . . . 52
6.2.2.2 Application on Muscle – 7 Kilograms Weight . . . 54
6.3 Model Predictive Control Development . . . 56
7 CONCLUSIONS AND FUTURE DEVELOPMENTS 57
7.1 Future Work . . . 57

II APPENDICES 59
A SCRIPTS AND ALGORITHMS 60
A.1 Scripts . . . 60
A.1.1 Stair Input Generation . . . 60
A.1.2 Reference Signal Generation . . . 60
A.2 Optimization Algorithms . . . 61
A.2.1 Genetic Algorithm . . . 61
A.2.1.1 Algorithm Initialization . . . 61
A.2.1.2 Population Evaluation . . . 62
A.2.1.3 Parameter Checking for New Individuals . . . 62
A.2.1.4 Mating Step . . . 63
A.2.1.5 Mating Partners Selection . . . 63
A.2.1.6 Crossover . . . 64

A.2.2 Modified Firefly Algorithm 64
A.2.2.1 Firefly Algorithm Parameter Update 65
A.2.3 Particle Swarm Optimization 65
A.2.3.1 Evaluation and Movement 65
A.2.3.2 Fit Checking for Particles 65
B THE BOUC-WEN MODEL OF HYSTERESIS 67

BIBLIOGRAPHY 69

LIST OF FIGURES

Figure 1 Geometric structure of a McKibben artificial muscle 7
Figure 2 Input waves for proportional valves 9
Figure 3 Pressure and displacement characteristic 10
Figure 4 Pressure and displacement characteristic 11
Figure 5 Transfer function fit 11
Figure 6 Comparison between experimental data and simulated output 16
Figure 7 Comparison between experimental data and simulated output with classic Bouc-Wen model 17
Figure 8 Comparison between experimental data and simulated output with generalized Bouc-Wen model 18
Figure 9 Genetic Algorithm – Classic Bouc-Wen Model (Random crossover) 32
Figure 10 Genetic Algorithm – Classic Bouc-Wen Model (Weighted crossover) 33
Figure 11 Genetic Algorithm – Classic Bouc-Wen Model (Proportional crossover) 34
Figure 12 Hysteresis loop result – Genetic Algorithm's parameters (classic Bouc-Wen model, proportional crossover approach) 35
Figure 13 Modified Firefly Algorithm – Classic Bouc-Wen Model 36
Figure 14 Hysteresis loop result using Modified Firefly Algorithm parameters (classic Bouc-Wen model) 37
Figure 15 Particle Swarm Optimization – Classic Bouc-Wen Model . . . 38
Figure 16 Hysteresis loop result using Particle Swarm Optimization parameters 39
Figure 17 Genetic Algorithm – Generalized Bouc-Wen Model (Random crossover) 41
Figure 18 Genetic Algorithm – Generalized Bouc-Wen Model (Weighted crossover) 42
Figure 19 Genetic Algorithm – Generalized Bouc-Wen Model (Proportional crossover) 43
Figure 20 Hysteresis loop result – Genetic Algorithm's parameters (generalized Bouc-Wen model, proportional crossover approach) . 44
Figure 21 Hysteresis loop result using Modified Firefly's parameters (generalized Bouc-Wen model) 45
Figure 22 Modified Firefly Algorithm – Generalized Bouc-Wen Model . 46
Figure 23 Hysteresis loop result using Particle Swarm Optimization parameters (generalized Bouc-Wen model) 47
Figure 24 Particle Swarm Optimization – Generalized Bouc-Wen Model 48
Figure 25 DS1104 R&D Controller Board 50
Figure 26 Block diagram of a PID controller in a feedback loop 51
Figure 27 Displacement control, 4.5 kilograms, 1 $^{rad}/_{s}$ 52
Figure 28 Displacement control, 4.5 kilograms, 2 $^{rad}/_{s}$ 53
Figure 29 Displacement control, 4.5 kilograms, 3 $^{rad}/_{s}$ 53
Figure 30 Displacement control, 7 kilograms, 1 $^{rad}/_{s}$ 54

Figure 31 Displacement control, 7 kilograms, 2 $^{rad}/_{s}$ 55
Figure 32 Displacement control, 7 kilograms, 3 $^{rad}/_{s}$ 55
Figure 33 MPC problem description example using CVXGEN 56
Figure 34 Graph force versus displacement for a hysteresis functional . 67

LIST OF TABLES

Table 1 Parameters for the Classic Bouc-Wen Model 16
Table 2 Parameters for the Generalized Bouc-Wen Model 17
Table 3 Parameters for the genetic algorithm 20
Table 4 Parameters for the genetic algorithm 26
Table 5 Parameters for the Genetic Algorithm approach – Classic Bouc-Wen model 31
Table 6 Random crossover – Final values (Classic Bouc-Wen model) . 32
Table 7 Weighted crossover – Final values (Classic Bouc-Wen model) . 33
Table 8 Weighted crossover – Final values (Classic Bouc-Wen model) . 34
Table 9 Parameters for the Modified Firefly Algorithm approach – Classic Bouc-Wen model 35
Table 10 Modified Firefly Algorithm – Final values (Classic Bouc-Wen model) 36
Table 11 Parameters for the Particle Swarm Optimization approach – Classic Bouc-Wen model 38
Table 12 Particle Swarm Optimization – Final values (Classic Bouc-Wen model) 39
Table 13 Parameters for the Genetic Algorithm approach – Generalized Bouc-Wen model 40
Table 14 Random Crossover – Final values (Generalized Bouc-Wen model) 41
Table 15 Weighted Crossover – Final values (Generalized Bouc-Wen model) 42
Table 16 Proportional Crossover – Final values (Generalized Bouc-Wen model) 43
Table 17 Parameters for the Modified Firefly Algorithm approach – Generalized Bouc-Wen model 45
Table 18 Modified Firefly Algorithm – Final values (Generalized Bouc-Wen model) 46
Table 19 Parameters for the Particle Swarm Optimization approach – Generalized Bouc-Wen model 47
Table 20 Particle Swarm Optimization – Final values (Generalized Bouc-Wen model) 48
Table 21 Timing performance comparison among algorithms 49
Table 22 PID gains for 4.5 kilograms 52

ACRONYMS

PAM Pneumatic Artificial Muscle

PID Proportional-Integrative-Derivative

MPC Model Predictive Control

EA Evolutionary Algorithm

GA Genetic Algorithm

FA Firefly Algorithm

MFA Modified Firefly Algorithm

PSO Particle Swarm Optimization

Part I

THESIS

1 INTRODUCTION

1.1 BACKGROUND AND MOTIVATION

The current state of the art in hydraulic actuators consists almost entirely of oil driven valves, pistons and motors. However, this kind of actuator cannot be used in some particular applications, such as power assist systems and rehabilitation: they have significant heaviness and rigidity, because of their mechanical structure and motorization [1]. In this context, it is problematic to share a robot working space with humans around it, and so this kind of actuator cannot be used to actuate, for example, orthotics.

McKibben muscles were invented by Joseph L. McKibben to motorize pneumatic art orthotics. They in general consist of an inner rubber tube enclosed in a braided outer nylon sleeve. These muscles can be used as actuators of rehabilitation systems due to the following advantages:

- Light weight
- High power to weight ratio
- High flexibility
- Low cost
- Low environmental impact

However, there are drawbacks: it is well known that the muscle has poor control performance due to the existence of strong nonlinearities, such as hysteresis, cavitation and saturation characteristics. Furthermore, the wear of the materials (nylon sleeve and rubber tube) may cause a shorter lifetime with respect to other actuators.

1.2 MODELING OF MCKIBBEN ARTIFICIAL MUSCLE

As already mentioned, the control of McKibben muscles is not easy to achieve. A PID control solution has been developed, although it has flaws: the parameters of the controller will have to be tuned different types of muscles and for various loads.

Model-based control techniques are better suitable for this job, but harder to develop. The plant model needs to be very precise for the control to be effective, and to do so identification techniques are used to get a first linear approximation of the model. Later, a hysteresis component is added to this linear model, to keep track of the nonlinearities added by the hysteretic behaviour of the muscle. Adding a hysteresis component to a linear model makes it more complex, so the choice of an appropriate hysteresis modeling technique is crucial to achieve good control performance.

This allows to get a model having good fit with respect to the real one. After this, it is possibly to try applying a model-based control approach, namely the **Model Predictive Control** (MPC). Further details about the development of controllers are in Chapter 6.

1.3 AIM OF THE STUDY

The aim of this dissertation is to get precise control of the displacement for a tap water-driven McKibben artificial muscle. To do so, a list of steps will be followed:

1. *Derivation of a simple model*

   Using linear identification techniques, it is possible to map the input pressure to the output displacement, and get a precise yet simple linear model of the muscle. While in pneumatic artificial muscles it is required to take into account the temperature dynamics and the compressibility of air, using water as the muscle allows to disregard them. This grants a simpler model of the muscle.

2. *Introduction of hysteresis component*

   The linear identification method grants a very simple yet linear model that does not take nonlinearities into account. The biggest source of nonlinearity for the McKibben muscle is hysteresis. A hysteresis component is added to the linear model, so that it will better follow the actual behaviour of the muscle. This step is essential to get good control performance.

3. *Hysteresis parameter estimation using optimization algorithms*

   Evolutionary algorithms and metaheuristic optimization can be used in order to find suitable parameters for the hysteresis model. This procedure is useful to grant a good fit between real and simulated data, so that it is possible to make use of a precise plant model in a model-based control.

4. *Controller Design*

   The application of Proportional-Integrative Control and Model Predictive Control techniques are proposed, using the muscle model obtained from linear system identification and modified with the introduction of the hysteresis component.

2 COMPOSITION OF THE THESIS

This dissertation is divided into chapters as follows.

Chapter 3 describes the McKibben artificial muscle, its general applications as well as the pros and drawbacks of using this kind of actuator.

Chapter 4 concerns the process of obtaining a simplified model from real data through model identification techniques. After this, a hysteresis component is introduced in the model, namely the Bouc-Wen model of hysteresis, to grant a better fit between simulated and real data. This step is essential so that this model can be used together with model-based control techniques so that it is possible to precisely control the muscle's position. Two different versions of the Bouc-Wen model have been studied and introduced, and their differences are highlighted in this Chapter.

Chapter 5 introduces the concept of *Evolutionary Algorithm* (EA) and *Metaheuristic Optimization* (MO). Three methods belonging to these categories of optimization techniques have been developed in order to find the best selection of parameters for the hysteresis model, and grant the best fit possible.

- **Evolutionary Algorithms** simulate the process of evolutio that characterizes biological systems.
  A *Genetic Algorithm* [2] (GA) approach has been studied and implemented to find a set of parameters that could provide a good fit between simulated and experimental values.
- **Metaheuristic Optimization** (MO) algorithm do not guarantee that a globally optimal solution may be found, but are designed to find, generate or select a set of parameters that may provide a sufficiently good solution to an optimization problem. Metaheuristics sample a set of solutions which is too large to be completely sampled. One of the main advantages of this method is that they may be used for a variety of problems with high performance.
  Two methods belonging to this category have been studied and developed: the *Firefly Algorithm* [3], which mimics the behaviour of fireflies, and *Particle Swarm Optimization* [4], which iteratively tries to improve a candidate solution with regard to a given measure of quality.

The performance of each algorithm, based both on calculation time and fit result, is compared in Sections 5.4.2, 5.4.3 and 5.4.4.

Chapter 6 covers the development and application of a PI control, as well as the tentative development of a model predictive control.

Chapter 7 draws the conclusions and gives mentions for future works.

A series of appendices follows the chapters: Appendix A contains all scripts and algorithms code used for this work, all thoroughly described. Appendix B contains a description of the standard Bouc-Wen model of hysteresis.

3

TAP WATER-DRIVEN MCKIBBEN ARTIFICIAL MUSCLE

4

MODELLING OF MCKIBBEN ARTIFICIAL MUSCLES

In this Chapter, the muscle models used as nominal models for model-based control are derived.

4.1 INTRODUCTION

McKibben muscles are actuators used mainly for medical purposes in rehabilitation and welfare. The reason is their high flexibility, light weight, low cost, human and environmental friendliness and ease of use. As mentioned in Chapter 1, the control of this kind of actuators is often problematic, due to their inherent high nonlinearity.

Many muscle models have been proposed, both static and dynamic. One of the most known static models is derived by Chou [5], which is based on the equilibrium between the input pressure and the release of energy. Although the main interest is to obtain the dynamic model to use in control, the static one is also reviewed and evaluated because they are used with feed-forward control applied to rehabilitation devices using the McKibben muscle.

Dynamic muscle models can be categorized in analysis oriented and control oriented. Analysis oriented models provide very high accuracy, but they are also very complex. For this reason they are not much suitable for control purposes.

Control oriented models provide lower accuracy than analysis oriented ones, but their lower complexity allows them to be used more efficiently for control.

Since tap water driven muscles are simpler than pneumatic ones, the idea is to use linear system identification and obtain a simple model of the muscle's dynamics. Being this only a linear model, it does not take into account the presence of strong nonlinearities, such as the friction between the braids and the hysteretic behaviour of the muscle. However, the introduction of a hysteresis model can lead to achieving higher precision with the model derived from linear system identification.

Through history, several hysteresis models have been developed. Notable examples are the Maxwell-slip model [6], the Jiles-Atheron model [7] and the Preisach model [8]. Common interest points of these models are friction of the braided sleeve, and the hysteresis caused by it, which both add nonlinearities to the model.

These hysteresis models are precise, yet complex in structure. To overcome this problem, the Bouc-Wen [9] [10] hysteretic model is combined with the identified muscle model. Including the hysteretic model raises the number of parameters that have to be identified for the system, but this is easily achieved, at first, by trial and error. Later, an adaptive algorithm is developed to get the best parameters for the muscle, and thus the updated model can be use as a nominal model for model-based control techniques.

The accuracy of the proposed model is compared to experimental results by analysing the pressure–displacement characteristics and the hysteresis characteristics. Moreover, the effects of changing the load of the muscle are studied, because they may have a great impact upon control performances.

4.2 STATIC MODEL

4.2.1 *Introduction*

The relationship between the axial contraction force and the pressure difference amid supply and atmospheric pressure has been reported in different papers [11] [12].

The association is based on the equilibrium between the input work in the McKibben muscle (i. e. when the fluid is supplied to the inner rubber tube) and the output work (i. e. when the actuator shortens or elongates because of the volumetric change associated with the pressure difference).

Figure 1 shows the geometric structure, which follows the geometric relationships in Equation 1.

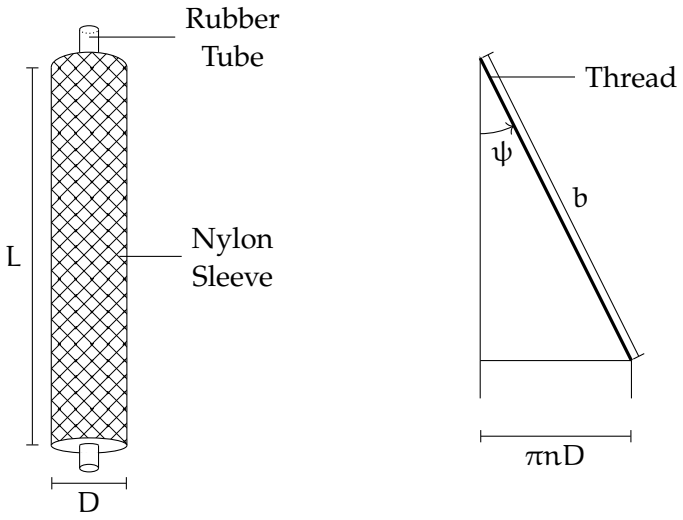

Figure 1: Geometric structure of a McKibben artificial muscle

$$L = b\cos(\psi), \quad D = \frac{b\sin(\psi)}{n\pi} \tag{1}$$

where L is the length of the muscle, D is the outer diameter of the muscle, b is the thread length, n is the number of turns of a thread, and ψ is the angle formed by the sleeve's threads and its vertical axis.

4.2.2 *Static Model of the Muscle*

The input work W_{in} is applied to the muscle when the fluid (liquid or air) pushes the internal surface of the rubber tube. This can be expressed as the product of the supply pressure and the change in volume.

$$dW_{in} = (P - P_0)\, dV = P' dV \tag{2}$$

Where P is the supply pressure, P_0 is the atmospheric pressure, P' is the pressure difference and dV is the volumetric change.

Equation 3 shows the volume of the muscle, with the assumption that it has cylindrical shape.

$$V = \frac{1}{4}\pi L D^2 \tag{3}$$

Then, from Equation 1,

$$V = \frac{1}{4}\pi L D^2 = \frac{b^3}{4\pi n^2}\sin(\psi)^2\cos(\psi) \tag{4}$$

When the actuator contracts, the output work of the system W_{out} is given by Equation 5.

$$dW_{out} = F \times (-dL) \tag{5}$$

Where F is the axial tension, and dL is the axial displacement change.
From Equation 1 we can express $dL/d\psi$ as follows.

$$\frac{dL}{d\psi} = -b\sin(\psi) \tag{6}$$

With the assumption that the input work and the output work should be equal if the system is lossless and without energy storage, as for Equations 2, 4, 5 and 6, then the relation shown in Equation 7 can be obtained.

$$F = -P'\frac{dV/d\psi}{dL/d\psi} = \frac{\pi P'}{4}\left(\frac{b}{\pi n}\right)^2\left(3\cos^2(\psi) - 1\right) \tag{7}$$

From this, it can be observed that the tension F is linearly proportional to the pressure difference P', and it is a monotonic function of the angle ψ ($0° < \psi < 90°$).

4.2.3 *Modified Static Model*

The static model of a McKibben muscle can be expressed as Equation 7. However, that model does not take into account the thickness t_k of both the nylon sleeve and the internal rubber tube. If this is considered, then the volume calculated in Equation 3 becomes the following.

$$\begin{aligned} V &= \frac{1}{4}\pi L\left(D - 2t_k\right)^2 \\ &= \frac{b^3}{4\pi n^2}\sin(\psi)\left(3\cos^2(\psi) - 1\right) - b\cos(\psi)t_k\left(\frac{b}{n}\sin(\psi) - \pi t_k\right) \end{aligned} \tag{8}$$

From Equation 8 then it is possible to obtain the equivalent of Equation 7 with thickness included. The result is shown in Equation 9.

$$F = \frac{b^2 P'}{4\pi n^2}\left(3\cos^2(\psi)\right) + \pi t_k P'\left[\frac{b}{\pi n}\left(2\sin(\psi) - \frac{1}{\sin(\psi)}\right) - t_k\right] \tag{9}$$

4.3 MUSCLE MODEL BASED ON SYSTEM IDENTIFICATION

If all sources of non linearity could be taken into account, then theoretical models may agree with experimental results. However, by taking into consideration these elements, the models would become very complex and they would be difficult to control. Moreover, the models would lack in versatility, meaning that these parameters would have to be changed for different muscles and weights.

The model of the muscle is thus obtained through linear system identification techniques and tools such as MATLAB and Simulink. In this section the derivation of the muscle model based on system identification is described, as well as the improvement of the identified model by using the Bouc-Wen model of hysteresis.

4.3.1 *Identification Experiments*

For the identification step, a stepwise signal has been generated and given to the input and output proportional valves. The two signals are symmetric one another, due to the nature of the valves. They range from 0 to 10 Volts, and are shown in figure 2. The algorithm for generating the signals is in Appendix A (Algorithm A.1.1).

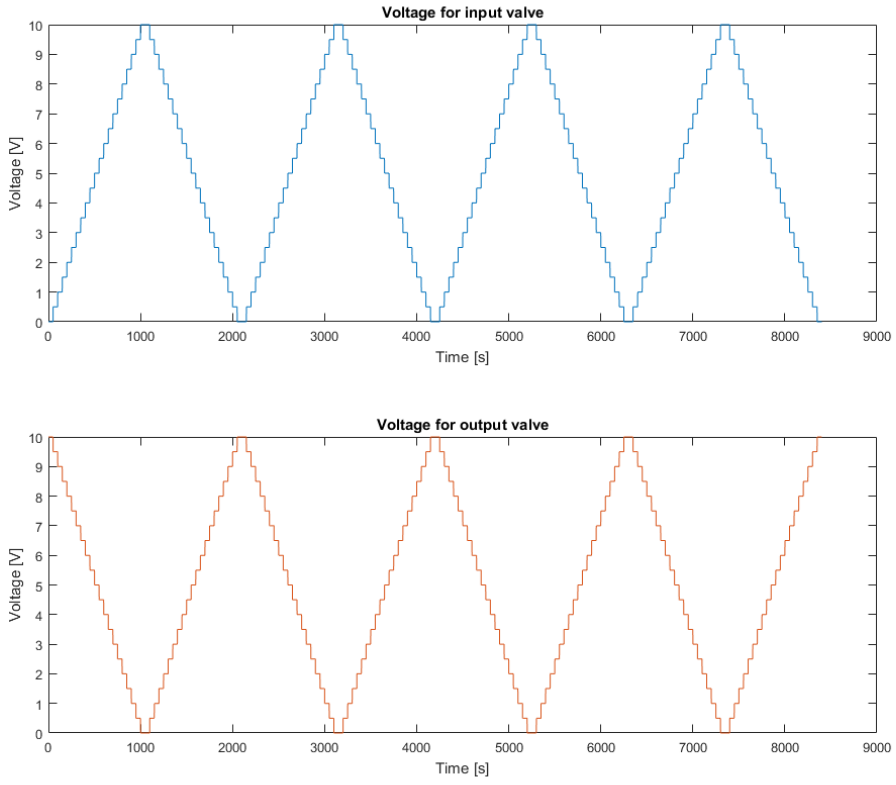

Figure 2: Input waves for proportional valves

Figure 3 shows the pressure and displacement characteristics of the muscle with the input just introduced.

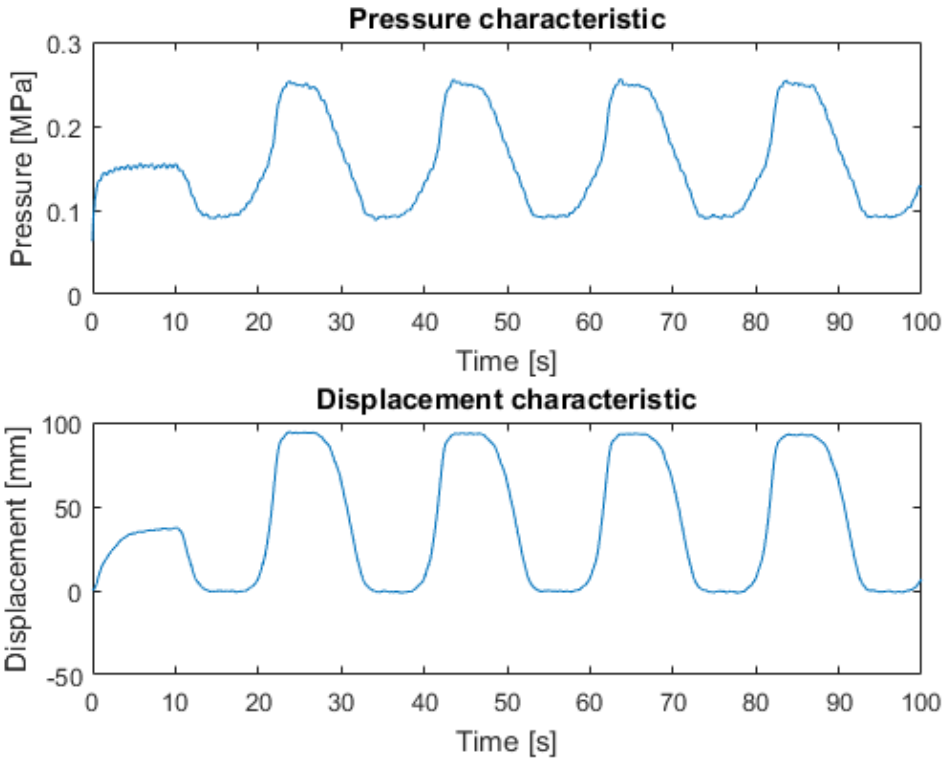

Figure 3: Pressure and displacement characteristic

By plotting this data with pressure on the x-axis and displacement on the y-axis, it is possible to notice the hysteretic behaviour of the muscle, as shown in Figure 4.

Using the MATLAB System Identification Toolbox [13], we can get a discrete transfer function from input and output values. The description on how to use the toolbox is provided in Appendix **??**.

The input and output data are given to the toolbox. After that, input and output data are treated in the following way:

- their mean is put to 0
- the data is split in half and used in the following way:
  - the first half is used for the *estimation process*, by which the toolbox attempts to find a discrete function that correctly approximates the input-output characteristic of the system.
  - the second half is used for the *validation process*, by means of which the toolbox computes the fit of the identified transfer function.

The toolbox permits to select the number of zeros and poles that the transfer function must present. A number of 1 poles and 1 zeros has been chosen for this step.

The identification step then provides the discrete transfer function in Equation 10, where L(z) and P(z) represent respectively the z transform of displacement and pressure.

$$G(z) = \frac{L(z)}{P(z)} = \frac{24.9366\, z^{-1}}{1 - 0.9618\, z^{-1}} \tag{10}$$

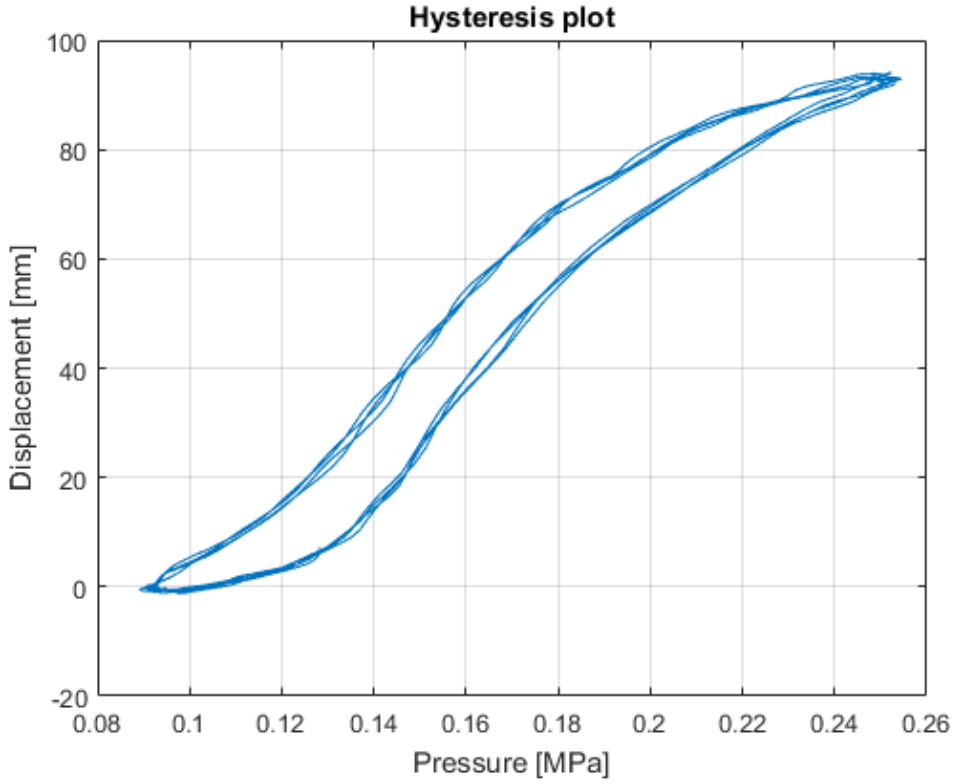

Figure 4: Pressure and displacement characteristic - Hysteresis plot

The fit of this transfer function with respect to experimental data is shown in Figure 5.

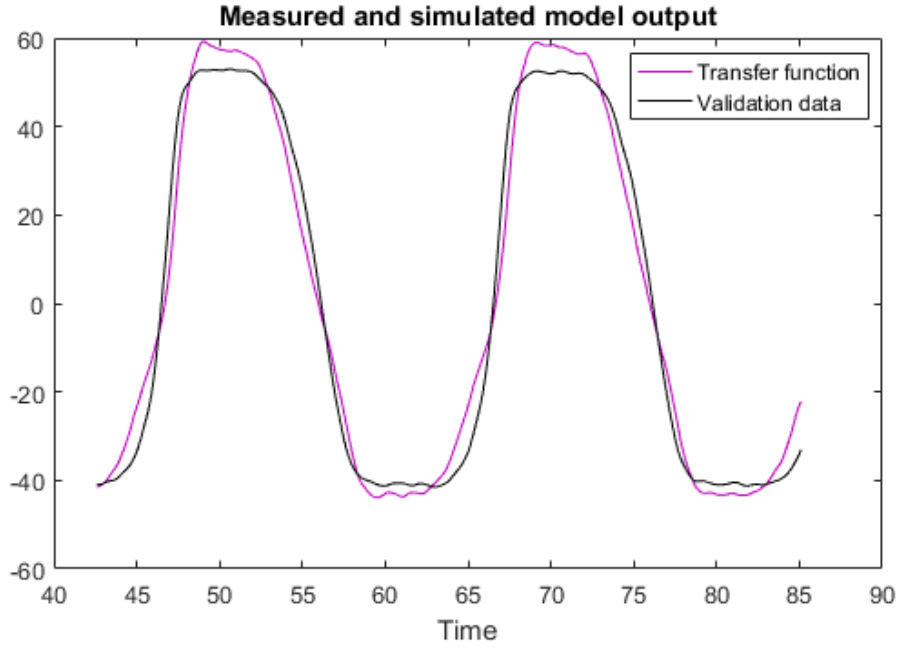

Figure 5: Transfer function fit

It is possible to raise the number of poles and zeros of the transfer function in order to improve its fit with respect to experimental data. The trade-off is that the resulting transfer function will be more complex.

The transfer function in Equation 10 can be written rewritten as follows

$$L(z) - 0.9618L(z)z^{-1} = 24.9366P(z)z^{-1} \tag{11}$$

This implies

$$\begin{aligned} l(k) &= 0.9618l(k-1) + 24.9366p(k-1) \\ &= al(k-1) + bp(k-1) \\ a &= 0.9618, \quad b = 24.9366 \end{aligned} \tag{12}$$

So the displacement at step k depends on the displacement and pressure at the previous step.

4.4 HYSTERESIS MODELING

Hysteresis represents a property of systems that show the dependence of their state from their history. It is a nonlinear phenomenon exhibited by systems stemming from various science and engineering areas: under a low-frequency periodic excitation, the relationship between the system's input and output is not the same for loading and unloading This means that the same input applied at different times can yield different output with respect to the history of the system.

The etymology of the term "hysteresis" is derived from the ancient Greek, and it means "delay" or "lagging behind".

A fundamental theory that allows a general framework for modeling hysteresis has not been developed yet.

For specific problems, models derived from a general understand of physics are normally used. This kind of models are, however, hard to derive and difficult to use in practical application due to their complexity. For this reason, alternative models of these complex systems have been proposed. Phenomenological (or semi-physical) models are simpler models that, although not giving the best description of the behaviour of a system, combine some physical understanding of the system along with some kind of black-box modeling, keeping relevant input-output information that is useful for characterization, design and control purposes.

4.5 MODELS OF HYSTERESIS

There is a large variety of hysteresis models available in literature, such as:

- the Preisach model [14], used to model electromagnetic hysteresis
- the Maxwell-Slip model [15], used for friction simulation and compensation,
- the Bouc-Wen model, extensively used in the areas of smart structures and civil engineering [10]

The latter consists of a first-order nonlinear differential equation that relates the input displacement to the output restoring force in a rate-independent hysteretic way [16].

4.6 THE BOUC-WEN MODEL OF HYSTERESIS

Proposed by Bouc [9] in 1971 and later modified by Wen [10] in 1976, the Bouc-Wen model has been vastly used to mathematically describe components and devices that present hysteretic behaviours, particularly within the areas of civil and mechanical engineering.

The differential equation consists of several parameters. By choosing them appropriately, it is possible to accommodate the response of the model to the real hysteresis loop [17]. The derivation of the model is available in Appendix B.

Equation 12 can be modified by adding a virtual hysteresis component. This should allow the hysteresis loop to adapt to the experimental values by changing the model's parameters.

$$\begin{aligned} l(k) &= a\left[\alpha l(k-1) + (1-\alpha)w(k-1)\right] + bp(k-1) \\ \alpha &\in [0,1], \quad a = 0.9618, \quad b = 24.9366 \end{aligned} \tag{13}$$

Here w is a virtual hysteresis variable introduced by the use of Bouc-Wen model. It is possible to implement two different versions of the hysteresis model:

- the one proposed by Bouc and modified by Wen, that will be henceforth called *classic* version
- a more recent and flexible version proposed by Song and Der Kiureghian, that will be called *generalized* version

4.6.1 *Classic Bouc-Wen Hysteresis Model*

The classic Bouc-Wen model's virtual hysteresis variable w is described by Equation 14.

$$\dot{w} = A\dot{l} - \beta\left|\dot{l}\right||w|^{n-1} - \gamma\dot{l}|w|^{n} \tag{14}$$

A, β, γ and n are Bouc-Wen model's parameters. They control the shape and the slope of the hysteretic loop. Further details are described in Appendix B.

4.6.2 *Generalized Bouc-Wen Hysteresis Model*

Song and Der Kiureghian [18] proposed a generalized Bouc-Wen model for highly asymmetric hysteresis that in certain cases shown higher flexibility and resiliency than the classic Bouc-Wen model.

The generalized Bouc-Wen model's virtual hysteresis variable w is described by Equations 15 and 16.

$$\dot{w} = \dot{l}\left[A - |w|\Psi(l,\dot{l},w)\right] \tag{15}$$

$$\begin{aligned} \Psi(l,\dot{l},w) = {} & \beta_1\,\mathrm{sgn}(\dot{l}w) + \beta_2\,\mathrm{sgn}(l\dot{l}) + \beta_3\,\mathrm{sgn}(lw) \\ & + \beta_4\,\mathrm{sgn}(\dot{l}) + \beta_5\,\mathrm{sgn}(w) + \beta_5\,\mathrm{sgn}(l) \end{aligned} \tag{16}$$

where $\beta_1, \ldots, \beta_6$ are fixed parameters.

4.7 HYSTERESIS VARIABLE APPROXIMATION

Equations 14 and 15 should be approximated to a discrete time equation, as Equation 13 is in discrete time. To do so, it is possible to use forward difference approximation method, as shown in Equations 17 and 18. T_s represents the sampling time for the approximation.

$$\frac{dx}{dt} \approx \frac{x(t+T_s)-x(t)}{T_s} = \frac{x(k+1)-x(k)}{T_s} \tag{17}$$

$$\frac{d^2x}{dt^2} \approx \frac{x(t+2T_s)-2x(t+T_s)+x(t)}{T_s^2} = \frac{x(k+2)-2x(k+1)+x(k)}{T_s^2} \tag{18}$$

4.7.1 *Classic Bouc-Wen Model*

In the classic version of the Bouc-Wen model, the virtual hysteresis variable w can be discretized as follows.

$$\begin{aligned}\frac{w(k+1)-w(k)}{T_s} = A\frac{l(k+1)-l(k)}{T_s} - \beta\left|\frac{l(k+1)-l(k)}{T_s}\right||w(k)|^{n-1} \\ -\gamma\frac{l(k+1)-l(k)}{T_s}|w(k)|^n\end{aligned} \tag{19}$$

Solving for $w(k)$ gives the result in Equation 20.

$$\begin{aligned}w(k) = A\left[l(k)-l(k-1)\right] - \beta|l(k)-l(k-1)||w(k-1)|^{n-1} \\ -\gamma\left[l(k)-l(k-1)\right]|w(k-1)|^n + w(k-1)\end{aligned} \tag{20}$$

4.7.2 *Generalized Bouc-Wen Model*

In the generalized version of the Bouc-Wen model, the virtual hysteresis variable w can be discretized as follows.

$$\frac{w(k+1)-w(k)}{T_s} = \frac{l(k+1)-l(k)}{T_s}\left[A-|w(k)|\Psi(l,w,k)\right] \tag{21}$$

$$\begin{aligned}\Psi(l,w,k) = \beta_1\,\mathrm{sgn}\left(\left[\frac{l(k+1)-l(k)}{T_s}\right]w(k)\right) \\ +\beta_2\,\mathrm{sgn}\left(\left[\frac{l(k+1)-l(k)}{T_s}\right]l(k)\right) \\ +\beta_3\,\mathrm{sgn}\left(l(k)w(k)\right)+\beta_4\,\mathrm{sgn}\left(\frac{l(k+1)-l(k)}{T_s}\right) \\ +\beta_5\,\mathrm{sgn}\left(w(k)\right)+\beta_6\,\mathrm{sgn}\left(l(k)\right)\end{aligned} \tag{22}$$

Solving Equation 21 for w(k) gives the result in Equation 23.

$$\begin{aligned} w(k) &= [l(k) - l(k-1)]\,[A - |w(k)|\Psi(l, w, k)] + w(k-1) \\ \Psi(l, w, k) &= \beta_1 \operatorname{sgn}([l(k) - l(k-1)]\, w(k)) \\ &\quad + \beta_2 \operatorname{sgn}([l(k) - l(k-1)]\, l(k)) \\ &\quad + \beta_3 \operatorname{sgn}(l(k) w(k)) + \beta_4 \operatorname{sgn}(l(k) - l(k-1)) \\ &\quad + \beta_5 \operatorname{sgn}(w(k)) + \beta_6 \operatorname{sgn}(l(k)) \end{aligned} \tag{23}$$

4.8 IDENTIFIED MODEL SUMMATION

The previous sections can be summarized with Equations 24 and 25.

4.8.1 *Classic Bouc-Wen Model*

$$\begin{cases} l(k) = a\,[\alpha l(k-1) + (1-\alpha) w(k-1)] + b p(k-1) \\ w(k) = A\,[l(k) - l(k-1)] - \beta |l(k) - l(k-1)| |w(k-1)|^{n-1} \\ \qquad -\gamma\,[l(k) - l(k-1)]\,|w(k-1)|^n + w(k-1) \\ \qquad \alpha \in [0, 1]\,, \quad a = 0.9618, \quad b = 24.9366 \end{cases} \tag{24}$$

4.8.2 *Generalized Bouc-Wen Model*

$$\begin{cases} l(k) = a\,[\alpha l(k-1) + (1-\alpha) w(k-1)] + b p(k-1) \\ w(k) = [l(k) - l(k-1)]\,[A - |w(k)|\Psi(l, w, k)] + w(k-1) \\ \Psi(l, w, k) = \beta_1 \operatorname{sgn}([l(k) - l(k-1)]\, w(k)) \\ \qquad +\beta_2 \operatorname{sgn}([l(k) - l(k-1)]\, l(k)) \\ \qquad +\beta_3 \operatorname{sgn}(l(k) w(k)) + \beta_4 \operatorname{sgn}(l(k) - l(k-1)) \\ \qquad +\beta_5 \operatorname{sgn}(w(k)) + \beta_6 \operatorname{sgn}(l(k)) \\ \qquad \alpha \in [0, 1]\,, \quad a = 0.9618, \quad b = 24.9366 \end{cases} \tag{25}$$

4.9 EVALUATION OF PROPOSED MODELS

The models previously proposed are tested and evaluated to estimate the error with respect to experimental data.

Without taking the Bouc-Wen model into consideration, given experimental input values p, using equation 10 the simulated output $\hat{l}$ with respect the experimental values l is shown in Figure 6.

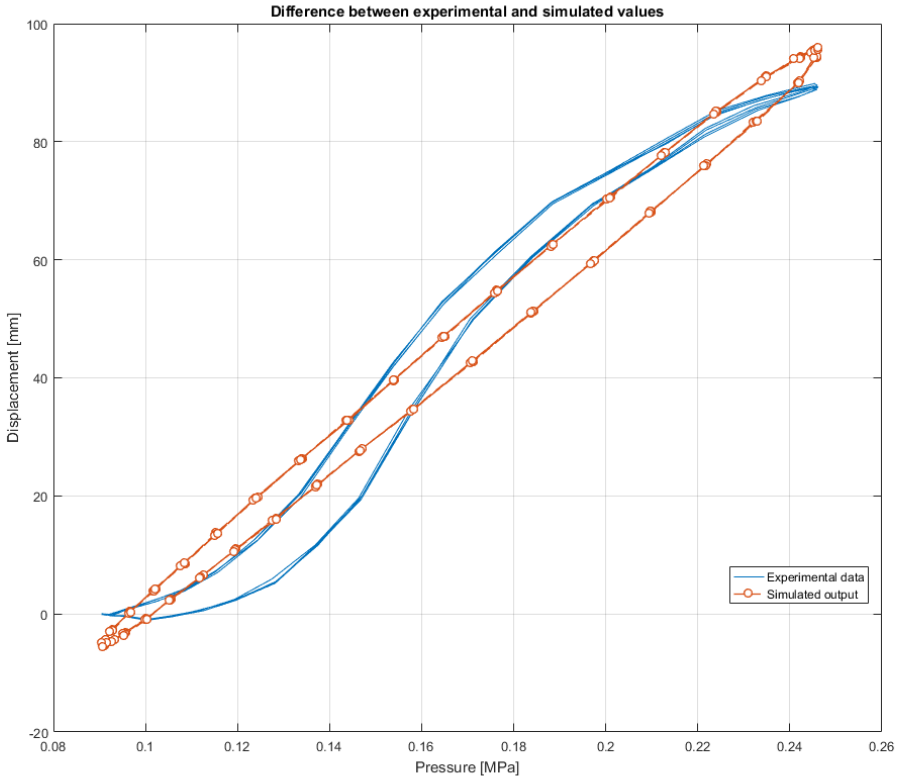

Figure 6: Comparison between experimental data and simulated output

If the Bouc-Wen model is taken into consideration, then the parameters must be fixed appropriately. To get a fairly good fit it is possible to tune these parameters by trial and error. Further methods for finding the best sets of parameters for the hysteresis model are discussed in Chapter 5.

4.9.1 *Classic Bouc-Wen Model Parameters*

The parameters to be chosen for the classic version of the Bouc-Wen model are α, A, β, γ and n. For this work, n will be fixed at 1 for both the classic and generalized version of the hysteresis model. Details about how each parameters alters the shape of the hysteretic loop are in Appendix B.
The parameters in Table 1 have been fixed by trial and error. All parameters are adimensional.

Parameter	Value
α	0.5
A	0.9
β	0.002
γ	-0.001
n	1

Table 1: Parameters for the Classic Bouc-Wen Model

Given experimental input p, the simulated output of the system taking into consideration the classic Bouc-Wen model of hysteresis is shown in Figure 7.

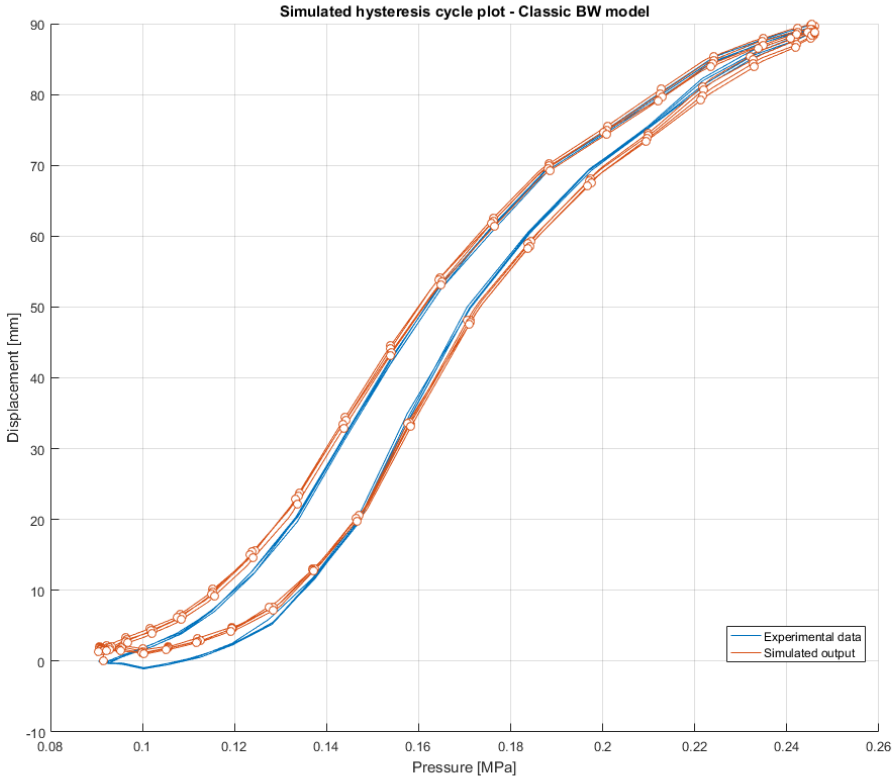

Figure 7: Comparison between experimental data and simulated output with classic Bouc-Wen model

4.9.2 *Generalized Bouc-Wen Model Parameters*

The parameters to be chosen for the generalized version of the Bouc-Wen model are α, A, n, $\beta_1, \ldots, \beta_6$. As previously stated, the value of n is fixed to 1.
As for the classic version of the hysteresis model, the parameters in Table 2 have been selected by trial and error. All parameters are adimensional.
Given experimental input p, the simulated output of the system after taking into consideration the generalized Bouc-Wen model of hysteresis is shown in Figure 8.

Parameter	**Value**
α	0.9
A	1
n	1
β_1	0.01
β_2	0.005
β_3	0.1
β_4	-0.01
β_5	-0.1
β_6	0.001

Table 2: Parameters for the Generalized Bouc-Wen Model

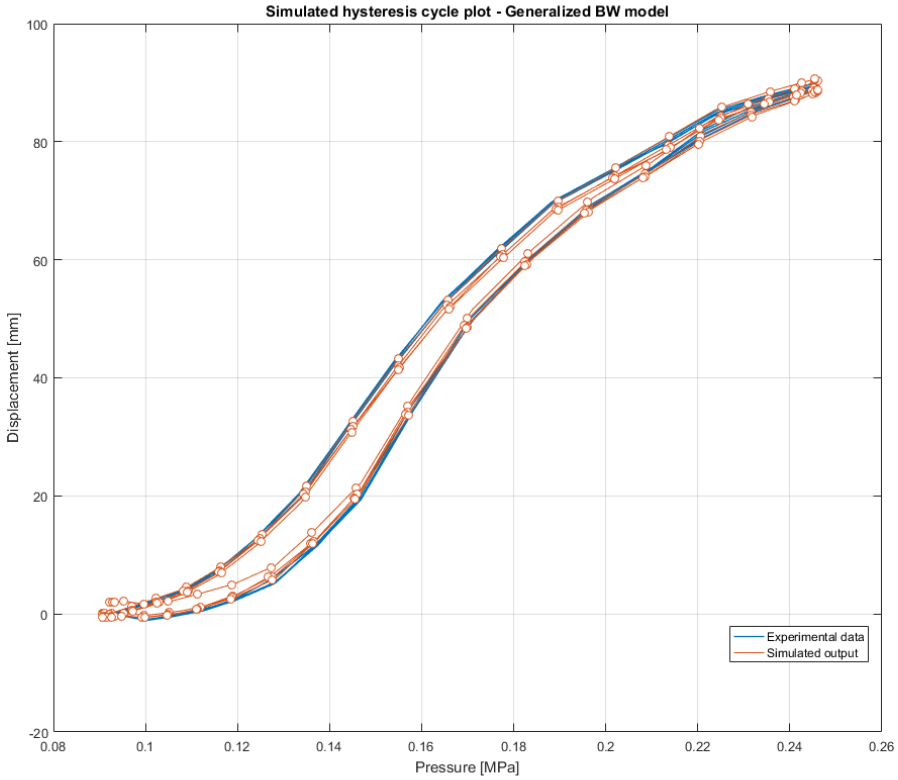

Figure 8: Comparison between experimental data and simulated output with generalized Bouc-Wen model

4.10 WEIGHT CHANGE TOLERANCE

Studies [19] [20] [21] have been done to verify how the model obtained for a specific muscle mutates as the load applied is changed.

The result is that the parameters of the Bouc-Wen model of hysteresis obtained for a specific weight can be used for other weights with little to no error in estimation. The results are not re-proposed nor replicated here as it would diverge from the purpose of this dissertation.

5 HYSTERESIS PARAMETER OPTIMIZATION

In this chapter, the methods to obtain the model's parameters are explained. As shown in Section 4.9, although parameters cherry-picked by trial and error proved successful to fit the approximated model onto experimental data, a higher grade of precision is desirable for applications demanding high accuracy.

Section 5.2 introduces the concept of *evolutionary algorithm*, with particular regard towards a simple genetic algorithm approach with the purpose of optimization.

After that, Section 5.3 introduces methods of *metaheuristic algorithm*, with the goal of transforming the search of parameters into an optimization problem.

Section 5.4 contains results and comparison for all algorithms previously described.

5.1 INTRODUCTION

After having described how to get a discrete model of the McKibben artificial muscle, the Bouc-Wen model of hysteresis has been introduced. This model proves useful to describe a hysteretic behaviour in a simple manner, and can provide great advantages in terms of performance when it comes to control.

However, there are parameters that have to be tuned in order for the model to operate correctly. The tuning may be done by trial and error, or the whole process may be automated.

The parameter refinement process makes use of an algorithm to find the parameters of the Bouc-Wen model that give a best approximation of the real experimental data. Such algorithms include, but are not limited to, Recursive Least Squares (RLS), evolutionary algorithms and multi-variable optimization algorithms.

For this work, different approaches based on evolutionary algorithms and multi-variable optimization methods have been tested.

5.2 EVOLUTIONARY ALGORITHMS

An evolutionary algorithm is a population-based optimization algorithm that mimics the behaviour of biological evolution, simulating the steps of reproduction, mutation, recombination and selection.

Each individual is tested in an optimization problem, and the best characteristic of the population are passed through the generations. The evaluation of an individual is done by means of a fitness function. A fitness function is a operator that assigns a fitness value to each individual. Depending on the nature of the problem, individuals with lower (or higher) fitness values are judged "better" from the algorithm and have a higher probability of being chosen in the mating step to generate new, better individuals. Therefore, their characteristics have a higher chance to be passed onto next generations in order to iteratively refine the parameters and arrive at a acceptable solution.

Parameter	Description
N	The population's size, i.e. the number of individuals (candidate solutions) that evolve to find a good approximation.
G	The maximum generation. After reaching that stage of evolution, the algorithm stops.
d	The chromosome's dimension, i.e. the number of variables to be optimized by the algorithm.
L	A d-dimensional vector containing the lower bounds for each variable.
U	A d-dimensional vector containing the upper bounds for each variable.
P_{cross}	The crossover probability. It controls the chance of inheriting a chromosome from a specific parent with respect to the other.
P_{mut}	The mutation probability.
m_m	The mutation magnitude, i.e. how much a given chromosome changes in the event of a mutation.

Table 3: Parameters for the genetic algorithm

After a certain amount of generations, or after one individual reaches a fixed fit value, the algorithm stops and the best individual is chosen as the solution to the optimization problem.

5.2.1 *Genetic Algorithm*

A genetic algorithm (GA) [2] is a kind of evolutionary algorithm inspired by the concept of natural selection that permeates a biological evolutionary process.

Genetic algorithms are a class of stochastic global search algorithm operating on a set called *population* of current approximations, called *individuals*. Individuals are encoded as a set (*chromosome*) of parameters (*genes*).

Once a population has been created, each individual performance is assessed according to the objective function characterising the problem that has to be solved.

Algorithms belonging to this class are divided into the following steps: *initialization*, *evaluation*, *crossover* and *mutation*. The general outline of the procedure is shown in Algorithm 1. The description and peculiarities of each step are discussed in Section 5.2.1.2. The genetic algorithm also makes use of several parameters, which are summarized in Table 3.

5.2.1.1 *Algorithm Overview*

5.2.1.2 *Algorithm Description*

5.2.1.3 *Initialization Step*

For a genetic algorithm, the initialization step consists in the creation of the individuals, and by extension that of the population.

Algorithm 1 Genetic algorithm approach

procedure GENETIC ALGORITHM
Input: N, G, d, L, U, p_{cross}, p_{mut}, m_m
Output: *A* d*-dimensional vector, i.e. the candidate solution (individual) with best fit value*
  Initialization: *generate* N d*-dimensional vectors* $n_i, i \in [1, N]$
  $current\_gen \leftarrow 1$
  while $current\_gen < G$ **do**
    Evaluation: *compute the fit value* $fit(i)$ *for each vector*
    Compute the normalized fit value $fit_n(i)$ *for each vector*
    $M_p \leftarrow$ *new empty Set*
    while $|M_p| < N$ **do**
      Crossover: *extract 2 individuals*
      Generate 2 new individuals (offspring)
      Add the offspring to M_P
    end while
    Mutation:
    for each $m \in M_p$ **do**
      for each *gene* $m_i, i = 1, \dots, d$ **do**
        Generate a random value r *such that* $0 \leqslant r \leqslant 1$
        if $r \leqslant P_{mut}$ **then**
          Mutate the i-th gene with percentage $\pm m_m\%$
  end while
  Return the individual with best fit value

One of the most common initialization methods is random initialization: each gene of any individual is randomly chosen between a defined interval. This gives a higher chance of finding good results from the first generations, and refine the solution starting from them.

Another initialization method may be to create a population of equal individuals, and mainly rely on mutation for the first generations in order to find better approximations. This method, while it may better reflect the natural behaviour of a biological evolutionary process, takes also a much larger number of generations to return appreciable results.

For this reason, the GA developed for this work makes use of random initialization. N d-dimensional vectors (individuals) p are created, following the constraints in Equation 26.

$$\begin{cases} p_i = [p_{i1}, \dots, p_{id}] \\ L_j \leqslant p_{ij} \leqslant U_j, \quad i = 1, \dots, N, \quad j = 1, \dots, d \end{cases} \tag{26}$$

5.2.1.4 *Evaluation Step*

The evaluation step concerns assessing a fitness value to each individual in order to rank them according to their performance in solving the optimization problem. The fitness value is assigned to each individual by means of a *fitness function*.

Experimental input pressure and output displacement, respectively described in Equations 27 and 28, are taken from data acquired by the identification experiments described in Section 4.3.1.

$$u = [u_1, u_2, \ldots, u_k] \tag{27}$$
$$l = [l_1, l_2, \ldots, l_k] \tag{28}$$

with k representing the number of points for which the experimental data has been sampled.

Each individual is used to perform a specific simulation of the model with its genes set as Bouc-Wen hysteresis parameters, using u as input. The simulated output displacement $\hat{l}$ is then compared to the experimental displacement l.

The fitness function chosen for this step is described in Equation 29. This function returns the fitness value for the i-th individual of the population.

$$\mathrm{fit}(i) = \frac{1}{\sum_{i=1}^{k} \left(l_i - \hat{l}_i\right)^2} \tag{29}$$

Judging from the nature of the fitness function, it is clear that individuals with a fitness value close to 0 are better than others, as it means that their simulated output is close to experimental data.

After that, the fit value of each individual is normalized in order to be used for the crossover step. To do so, each fit value is divided by the sum of all fit values, as shown in Equation 30.

$$\mathrm{fit}_n(i) = \frac{\mathrm{fit}(i)}{\sum_{i=1}^{n} \mathrm{fit}(i)} \tag{30}$$

It is important to notice that given the way that fit_n is constructed, the relation of Equation 31 arises.

$$\sum_{i=1}^{N} \mathrm{fit}_n(i) = 1 \tag{31}$$

5.2.1.5 *Crossover Step*

The crossover step serves the purpose of creating two new individuals (*offspring*) starting from two *parents*. The characteristics of the offspring will thus largely depend on their parents' ones.

Three different crossover policies have been developed for this purpose:

- **Random crossover**
  The random method assigns the genes from parents to offspring in a random fashion.
  Given a number x with $1 \leqslant x \leqslant d$, the offspring inherits the first x genes from the first parent and the remaining ones from the second parent. This process is repeated for the second offspring.

- **Weighted crossover**
  The weighted method makes so that the genes of the parent with higher fit have a higher probability of being chosen for the offspring.
  Given the fit values of the parents $\mathrm{fit}(p_1)$ and $\mathrm{fit}(p_2)$, with $\mathrm{fit}(p_1) \geqslant \mathrm{fit}(p_2)$, then the *inheritance probability* In_p is defined as follows:

  $$\mathrm{In}_p = \frac{\mathrm{fit}(p_1)}{\mathrm{fit}(p_1) + \mathrm{fit}(p_2)}$$

  For each offspring and gene, the probability of inheriting the given gene from the first parent will be In_p. Otherwise, the gene will be inherited from the second parent.
- **Proportional crossover**
  The proportional method creates the offspring's genes from the parents' ones, proportionally inheriting them based on their fit values.
  Given the fit values of the parents $\mathrm{fit}(p_1)$ and $\mathrm{fit}(p_2)$, with $\mathrm{fit}(p_1) \geqslant \mathrm{fit}(p_2)$, then the *proportion variables* pv_1 and pv_2 are defined as follows:

  $$pv_1 = \frac{\mathrm{fit}(p_1)}{\mathrm{fit}(p_1) + \mathrm{fit}(p_2)}, \quad pv_2 = 1 - pv_1$$

  Given $Gp_1(x), Gp_2(x), Go(x), 1 \leqslant x \leqslant d$ respectively for the first parent's, the second parent's and the offspring's x-th gene, then the following relation arises:

  $$Go(x) = Gp_1(x) \cdot pv_1 + Gp_2(x) \cdot pv_2$$

  This means that the offspring will be more similar to the parent with higher fit, while still carrying on some of the less performing parent's characteristics.

5.2.1.6 *Mutation Step*

The mutation step's purpose is to randomly mutate some genes to reflect the random behaviour of biological evolution through time.

For each gene $Go(x), 1 \leqslant x \leqslant d$ of each offspring generated, a random value r between 0 and 1 is generated. If this value is less than the *mutation probability* P_{mut}, then the gene is modified by a percentage of its value.

The final value $Go(x)'$ of the x-th gene will be:

$$Go(x)' = Go(x) \cdot (1 \pm m_m)$$

Here the *mutation magnitude* is added or subtracted with probability 0.5, by generating a new random value and verifying whether it surpasses this threshold.

5.3 METAHEURISTIC ALGORITHMS

A *metaheuristic* is a procedure designed to find a heuristic that may provide a sufficiently good solution to an optimization problem.

Metaheuristic algorithms do not guarantee that a globally optimal solution can be found on some class of problems, but they often provide a solution with a good fit value by exploring the space of possible solutions.

For this work, two metaheuristic algorithms have been developed and tested with the purpose of finding good parameters for fitting the Bouc-Wen hysteresis parameters of the identified transfer function over real experimental data.

The first is the *Firefly Algorithm*, described in Subsection 5.3.1. This algorithm mimics the movement of a swarm of fireflies inside the solution space, simulating an *attractiveness component*, governed by their *luminosity*, that pushes the swarm towards local and global optimal solutions. Two versions of this algorithm have been developed in order to grant a higher versatility for finding the parameters, depending on the required application. The modified version is described thoroughly in Section 5.3.2.

The second algorithm is the *Particle Swarm Optimization*, described in Section 5.3.3. It is a computational method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. It solves a problem by having a population of candidate solutions (*particles*) moving around in the search-space according to simple laws governed by their position and velocity. Each particle's movement is influenced by its local best known position, but is also guided toward the best known positions in the search-space. This value is updated whenever a better position is found any particle. This is expected to move the swarm toward the best solutions. The algorithm has been conceived as a mathematical representation of the movement of organisms in a bird flock or fish school.

The values chosen for algorithms' parameters are discussed in Section 5.4, which is dedicated to show the results and compare these algorithms.

5.3.1 *Firefly Algorithm*

The Firefly Algorithm (FA) is a metaheuristic optimization algorithm developed by Xin-She Yang [3] that mimics the behaviour of fireflies.

The main concept of such algorithm is to create a population of fireflies representing candidate solutions in a d-dimensional space for a certain optimization problem, where d is the number of variables to be optimized.

The fireflies are then attracted to each other following a set of rules:

- **Unisexuality**
  All fireflies are unisex, so that one firefly is attracted to other fireflies regardless of their sex.
- **Attractiveness**
  The attractiveness of a firefly is proportional to its brightness, thus for any two fireflies the less bright will move towards the brighter one. Both brightness and attractiveness decrease as the distance between two fireflies increases.
- **Brightness**
  The brightness of a firefly is determined and affected by the objective function.

Algorithm 2 Firefly Algorithm Approach

procedure FIREFLY ALGORITHM
Input: N, G, d, L, U
Output: *A* d*-dimensional vector, i.e. the candidate solution (firefly) with best fit value*
  Initialization: *generate* N d*-dimensional vectors* $n_i, i \in [1, N]$
  Light intensity I_i for n_i *is determined by* $f(n_i)$
  Define light absorption coefficient σ
  $current\_gen \leftarrow 1$
  while $current\_gen < G$ **do**
    for $i = 1, \ldots, N$ **do**
      for $j = 1, \ldots, N$ **do**
        if $I_i < I_j$ **then**
          Move firefly i towards j
          Vary attractiveness with distance r
          Evaluate new solutions and update light intensity
        end for j
      end for i
      Rank the fireflies and find the current global best g^*
  end while

5.3.1.1 *Algorithm Overview*

5.3.1.2 *Algorithm Description*

The following sections serves the purpose of describing in depth the steps and the specifics of the Firefly Algorithm.

Section 5.3.1.3 describes how light intensity influences the relative attractiveness between two fireflies.

Section 5.3.1.4 describes how fireflies move in the solution space through the iterations.

Table 4 summarizes and describes the parameters used by the algorithm.

5.3.1.3 *Light Intensity and Attractiveness*

The brightness I of a firefly at a particular location depends on the fitness value returned by the fitness function at that specific location.

Each firefly follows a path that brings it closer to other attractive fireflies. Relative attractiveness Λ_{ij} however is relative, and it will vary with the distance r_{ij} between firefly *i* and *j*.

Equation 32 shows the relation between distance and attractiveness, where Λ_0 is the attractiveness at zero distance and σ is a parameter called *light absorption coefficient*.

$$\Lambda_{ij} = \begin{cases} \Lambda_0 e^{-\sigma r_{ij}^2} & \text{if } I_j > I_i \\ 0 & \text{otherwise} \end{cases} \tag{32}$$

Equation 32 suggests that the brighter is a firefly, the more attractive it is to any other firefly. The light intensity I_j for the *j*-th firefly is determined by the fit function.

Parameter	Description
N	The population's size, i.e. the number of fireflies (candidate solutions) that move in the solution space to find a good solution.
G	The maximum generation. After reaching that stage of evolution, the algorithm stops.
d	The chromosome's dimension, i.e. the number of variables to be optimized by the algorithm.
L	A d-dimensional vector containing the lower bounds for each variable.
U	A d-dimensional vector containing the upper bounds for each variable.
Λ_0	Attractiveness at zero distance, described in Section 5.3.2.1
σ	Light absorption coefficient that influences the attractiveness distance, described in Section 5.3.2.2.
ρ	Randomization parameter influencing the fireflies' motion, described in Section 5.3.2.3

Table 4: Parameters for the genetic algorithm

$$I_j = f_{fit}(n_j) = f_{fit}(n_{j,1}, n_{j,2}, \ldots, n_{j,d}) \tag{33}$$

The fit function chosen for this work is the same as for the Genetic Algorithm approach, and is shown in Equation 29. The distance r_{ij} between firefly i and j at step k is given by the Euclidean distance between the two.

$$\begin{cases} r_{ij}^k &= \left\| n_j^k - n_i^k \right\| \\ &= \sqrt{(n_{j,1}^k - n_{i,1}^k)^2 + (n_{j,2}^k - n_{i,2}^k)^2 + \ldots + (n_{j,d}^k - n_{i,d}^k)^2} \end{cases} \tag{34}$$

5.3.1.4 *Firefly Movement*

The movement of a firefly n_i towards a more attractive n_j at step $k \in [1, G]$ is given by Equation 35.

$$n_i^k = n_i^k + \Lambda_{ij}^k \left(n_j^k - n_i^k\right) + \rho v_i^k \tag{35}$$

Here Λ_{ij} is the attractiveness of firefly j to firefly i, ρ is a *randomization parameter* and v_i is a vector of d random numbers drawn from a Gaussian or uniform distribution.

It is important to notice that the movement of a given firefly towards a less attractive one is given only by a random factor, but it is still present. This means that the most attractive fireflies still move and can potentially find better solutions.

After the fireflies' positions are updated, new fitness values are calculated. New generations are obtained by using the same set of equations. Over successive generations, the fireflies converge to local and global optimal values, completing the optimization process.

5.3.2 *Modified Firefly Algorithm*

A modified version of the Firefly Algorithm has been developed and studied by M.A. Zaman and U. Sikder [22]. The main idea behind the Modified Firefly Algorithm is to change the *process control parameters* Λ_0, σ and ρ through the iterations. Typical values of these parameters are $\Lambda_0 = 1$, $\sigma = 1$ and $\rho \in [0.1, 0.2]$. By changing these from static to dynamic parameters it is possible to perform a more precise tuning of the fireflies behaviour, reflecting thus on the performance of the algorithm on finding their optimal values.

5.3.2.1 *Attractiveness at Zero Distance*

It has been studied that the attractiveness at zero distance parameter Λ_0 controls the *exploration distance* of the fireflies: higher values of Λ_0 encourage global exploration while lower values encourage close refinement (local exploitation).

Hence it would be better to make Λ_0 a decreasing value over the iterations, such that initially the fireflies explore a wide range of possible solutions minimizing the risk of getting stuck in a local optimum, while near the end of the execution a lower value would allow for fine refinement around the different optima found.

The formula for the attractiveness parameter for iteration $k \in [1, G]$ has been changed as shown in Equation 36.

$$\Lambda_0(k) = a - b\frac{1-k}{1-G} \tag{36}$$

From Equation 36 it is possible to notice that the value of Λ_0 goes from a (when $k = 1$) to $a - b$ (when $k = G$).

5.3.2.2 *Light Absorption Coefficient*

The value of σ control the effect of distance on the motion of fireflies. Equation 32 suggests that lower values of σ will cause movement towards other attractive fireflies even if they are far away. This may lead to a premature convergence of a high number of fireflies towards a local maximum.

However, after a sufficient number of generations, it is desirable that the fireflies start to converge towards the maximum value found so far. To do so, a lower value of σ is required.

The formula for the *light absorption coefficient* σ for iteration k has been changed as shown in Equation 37.

$$\sigma(k) = 2 \cdot \left(7^{-k/G}\right), \quad c \geqslant 0, d \geqslant 0 \tag{37}$$

5.3.2.3 *Randomization Parameter*

The *randomization parameter* ρ controls the random component of the firefly's motion. A large value of ρ encourages broad random exploration. Using the same arguments as for the *light absorption coefficient*, it can be inferred that a gradual decrease of the value of ρ over successive iterations should improve the performance of the algorithm as the fireflies must explore the solution space widely at the beginning to find good candidate solutions and then refine them in latter steps.

The formula for the *randomization parameter* ρ for iteration k has been changed as shown in Equation 38.

$$\rho(k) = 0.1 \cdot e^{-4.8k/G} \tag{38}$$

The proposed modification of the algorithm using dynamic parameters are expected to provide a good mix of global exploration and local exploitation, resulting in high accuracy and fast convergence of the algorithm towards the global optimum.

5.3.3 *Particle Swarm Optimization*

Particle Swarm Optimization (PSO) is an evolutionary computation technique developed by Kennedy and Eberhart in 1995 [4] [23].

PSO is similar to a genetic algorithm, meaning that both algorithms are initialized with a population of random solutions. Unlike a GA, however, candidate solutions, called *particles*, are also assigned a randomized velocity. After that, they are flown into the problem space.

Each point of the problem space represents a set of parameters for the optimization problem that has to be solved, in this case the Bouc-Wen model parameter search Each particle keeps track of its coordinates in the problem space, and remembers the coordinate of the point which has yielded the best solution, in other words the point that has the highest fitness value found by them so far. This value is called *pbest*, which means *personal best*.

Another value is tracked and known by all particles, which is the *global best* value found by any particle up until that certain iteration of the algorithm. This value is called *gbest*.

The particle swarm optimization concept consists of, at each time step, changing the velocity (accelerating) each particle towards its *pbest* and *gbest* locations. Acceleration is weighted by a random term, with separate random numbers generated to dictate acceleration towards *pbest* and *gebest* locations.

5.3.3.1 *Algorithm Overview*

Algorithm 3 Particle swarm optimization approach

procedure PARTICLE SWARM OPTIMIZATION
Input: N, G, d, L, U, V_{max}
Output: *A* d*-dimensional vector, i.e. the candidate solution (particle) with best fit value*
  Initialization: *generate* N d*-dimensional vectors* $n_i, i \in [1, N]$
  $\texttt{current\_gen} \leftarrow 1$
  while $\texttt{current\_gen} < G$ **do**
    Evaluation: *compute the fit value* $\texttt{fit}(i)$ *for each vector*
    Pbest comparison: *Compare the particle's fit value with particle's* $\texttt{pbest}$*. If the current value is better than* $\texttt{pbest}$*, then set* $\texttt{pbest}$ *value equal to the current value, and* $\texttt{pbest}$ *location equal to the particle's current location in the* d*-dimensional space*
    Gbest comparison: *Compare the particle's fit value with* $\texttt{gbest}$*. If the current value is better than* $\texttt{gbest}$*, then set* $\texttt{gbest}$ *value equal to the current value, and* $\texttt{gbest}$ *location equal to the particle's current location in the* d*-dimensional space*
    Update: *change the particle's velocity and position*
  end while

5.3.3.2 *Algorithm Description*

The following sections serve the purpose of describing in depth the steps and the specifics of the Particle Swarm Optimization algorithm.

Section 5.3.3.3 describes how the values for velocity and position for the particles are updated each iteration.

Section 5.3.3.4 describes the purpose of and how to compute the inertia weight w used when calculating the speed of a particle.

Section 5.3.3.5 describes the purpose of the acceleration constants c_1 and c_2.

5.3.3.3 *Velocity and Position Update*

Given x_i and v_i which are respectively the i-th particle's position and velocity, then the movement of the i-th particle is dictated by Equations 39 and 40. In Equation 39, w is the *inertia weight*, x_{pb} is the position associated to the particle's personal best value, and x_{gb} is the position associated to the global best value among all particles.

$$v_i = w \cdot v_i + c_1 \cdot \texttt{rand}() \left(x_{pb} - x_i\right) + c_2 \cdot \texttt{rand}() \left(x_{gb} - x_i\right) \tag{39}$$

$$x_i = x_i + v_i \tag{40}$$

Particles' velocity are constrained by the maximum velocity parameter V_{max}. This is a very important parameter, as it determines the fineness with which regions between the present position and the best positions known are searched. If V_{max} is too high, particles my overshoot good solutions. On the other hand, if V_{max} is too low, particles may move very slowly and may not be able to find a good solution before the algorithm comes to a halt.

5.3.3.4 *Inertia Weight*

The use of a *inertia weight* w has provided improved performance in a number of applications, as an opportune selection of the inertia weight provides a balance between global and local exploration, and results in fewer iterations on average to find a sufficiently good solution.

The *inertia weight* w changes dynamically with the current iteration of the algorithm according to the following law:

$$w(k) = a - (a - b) \cdot \frac{k}{G}, \quad k \in [1, G]$$

5.3.3.5 *Acceleration Constants*

The *acceleration constants* c_1 and c_2 represent the importance of the personal and global best according to the particles. If c_1 is greater than c_2, the particles will tend to judge as better their own personal best and will be inclined to move towards that location.

5.4 ALGORITHM COMPARISON

Simulations have been run on all the algorithms proposed in the previous sections. Each one of them has been tested with both the standard and the generalized Bouc-Wen model of hysteresis. The results, for both models and for each algorithm, are respectively in Section 5.4.2 and 5.4.3.

Since the generalized version of the hysteresis component consists of 8 variables it is safe to assume beforehand that the execution time of the algorithms would be higher. A comprehensive comparison of the execution times for the algorithms is described in Section 5.4.4.

5.4.1 *System Specifications*

5.4.2 *Solution Comparison – Classic Bouc-Wen Model*

This Section serves the purpose of comparing the solutions found using the algorithms previously described when trying to find suitable parameters for the classic Bouc-Wen model of hysteresis.

5.4.2.1 *Genetic Algorithm – Classic Bouc-Wen Model*

Table 5 contains the parameters used for the genetic algorithm approach. Those parameters have been used for the genetic algorithm using all three kinds of crossover strategies: random, weighted and proportional.

Figures 9, 10 and 11 show the trend of the classic Bouc-Wen model parameters A, α, β and γ depending on the crossover strategy used [1].

It is important to note that due to how the three strategies have been developed, the performance of the random and weighted methods depend strictly on the initial

1 Figure 9 shows, for each generation, the parameters of the best individual, i.e. the individual with the highest fit value. This will be the standard way of presenting trends from now on.

Parameter	Value
N	30
G	30
d	4
L	$[0.5, 0.5, -0.5, -0.1]$
U	$[1, 0.99, 0.5, 0.1]$
P_{cross}	0.5
$Pmut$	0.1
m_m	0.05

Table 5: Parameters for the Genetic Algorithm approach – Classic Bouc-Wen model

population, as the individuals' chromosomes are always only swapped. The final result will approach the best possible solution attainable using the chromosomes of the initial population [2]. However, it is certainly possible that the final fit may not be acceptable.

Due to this fact, the proportional crossover strategy proved to be the best as it generates new chromosomes (and thus, completely new individuals) each generation. Regardless, a decreasing trend in each parameter is to be expected due to how the proportional strategy has been developed.

2 Apart from eventual changes introduced in case of mutations.

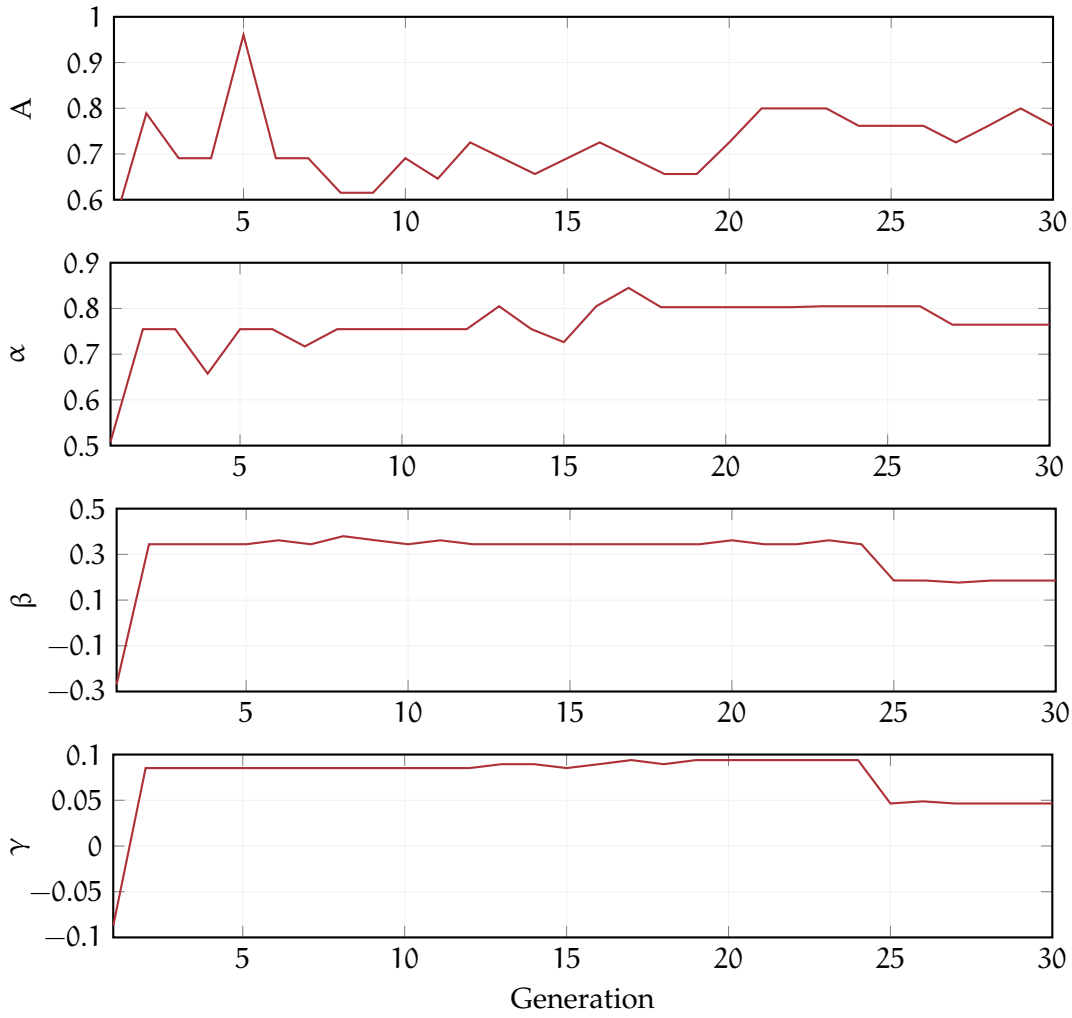

Figure 9: Genetic Algorithm – Classic Bouc-Wen Model
(Random crossover)

Parameter	A	α	β	γ
Final value	0.76158	0.76455	0.18539	0.04647

Table 6: Random crossover – Final values (Classic Bouc-Wen model)

A series of sharp changes is noticeable in Figure 9. As said beforehand, this behaviour is due to the fact that the set of chromosomes obtainable by the individual in the population is fixed to the initial one.

Nonetheless it is possible to notice that the variables β and γ are the first to settle to fairly consistent values, up until generation 25. This behaviour is probably due to the current best individual being near a local optimum.

The final values for each parameter using this crossover strategy are in Table 6.

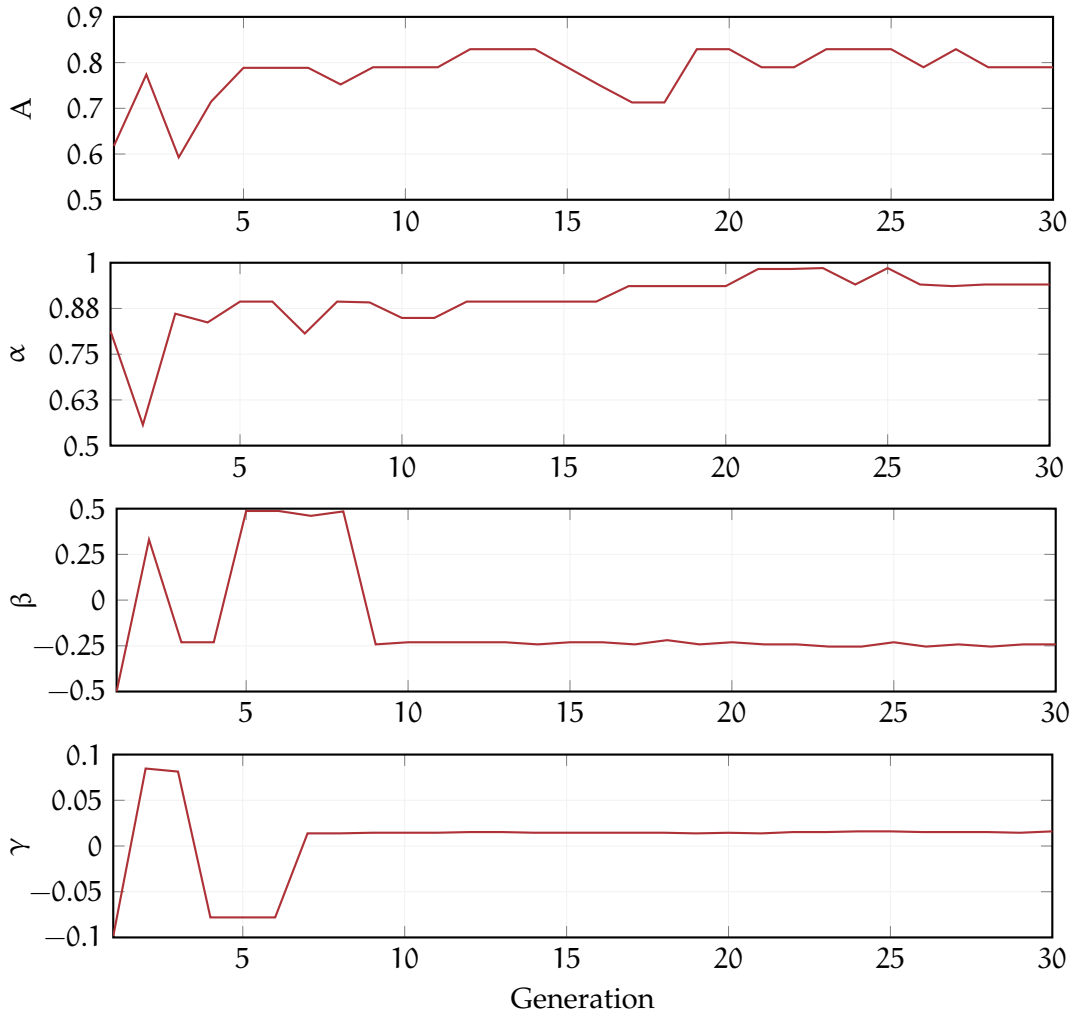

Figure 10: Genetic Algorithm – Classic Bouc-Wen Model (Weighted crossover)

As for the random crossover strategy, the weighted crossover strategy operates on the closed set of initial chromosome values, but combines them judging the fit value of the individuals instead of doing it randomly.

The most important difference between these two results is the fact that it takes fewer generations to reach what may be a global optimum. This is especially noticeable in Figure 10, as the parameters β and γ stabilize on similar values around generation 10 and 7, respectively.

The final values for each parameter using this crossover strategy are in Table 7.

Parameter	A	α	β	γ
Final value	0.78982	0.94050	−0.24214	0.01601

Table 7: Weighted crossover – Final values (Classic Bouc-Wen model)

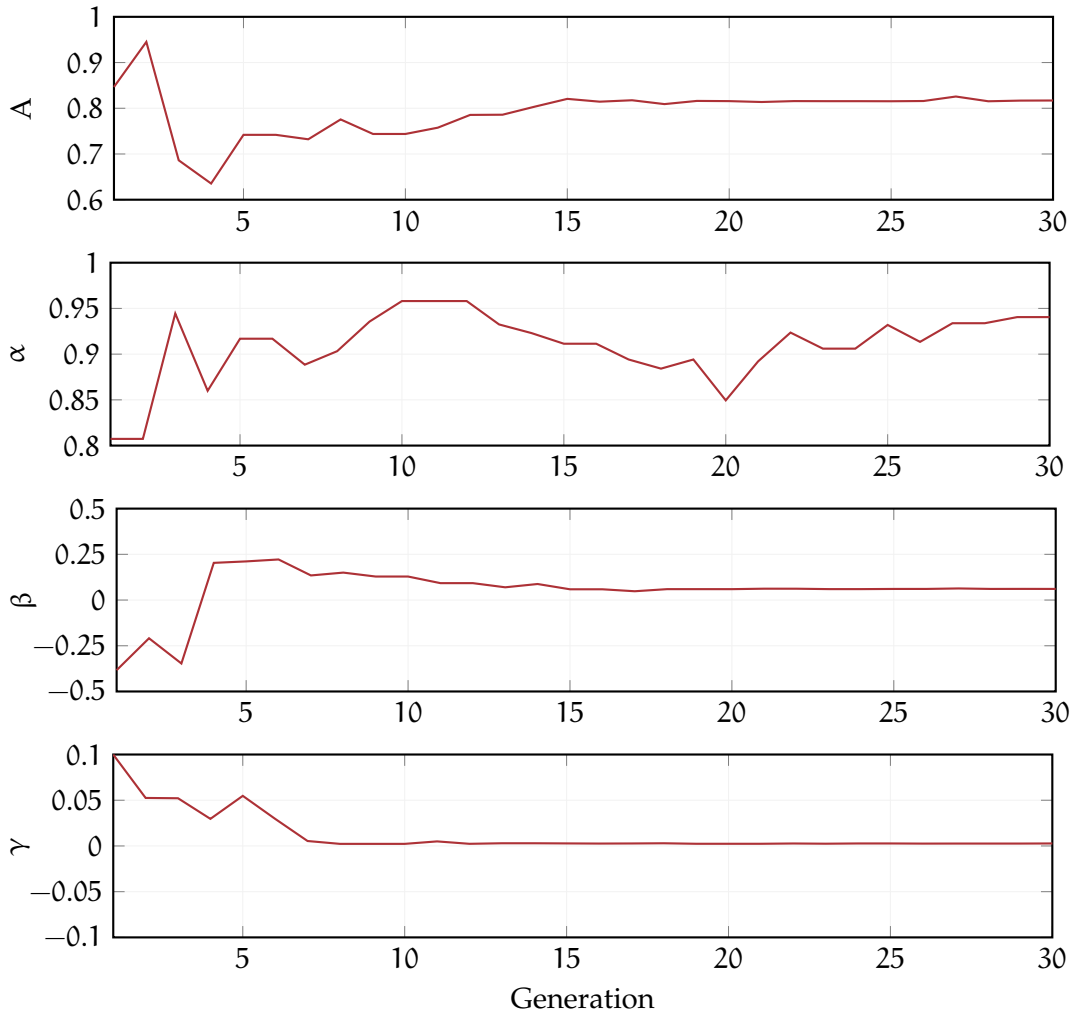

Figure 11: Genetic Algorithm – Classic Bouc-Wen Model (Proportional crossover)

In contrast to the parameter trends of random and weighted crossover strategies, the proportional crossover strategy shows a smoother trend. This is especially noticeable for parameters A, β and γ.

As for the weighted crossover method, it is possible to see an almost immediate setting of parameters β and γ, as it happens for both around generation 7.

The final values for each parameter using this crossover strategy are in Table 8.

Parameter	A	α	β	γ
Final value	0.81699	0.94050	0.06078	0.00273

Table 8: Weighted crossover – Final values (Classic Bouc-Wen model)

Figure 12 shows how the hysteresis loop behaves with the final parameters present in Table 8.

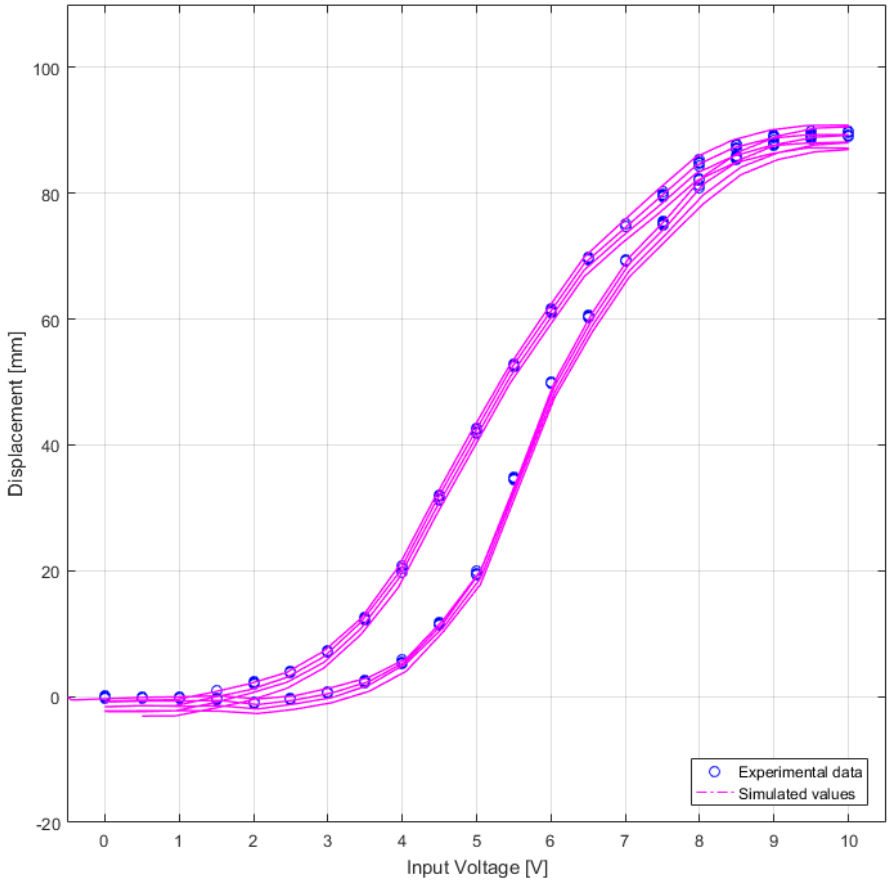

Figure 12: Hysteresis loop result – Genetic Algorithm's parameters (classic Bouc-Wen model, proportional crossover approach)

5.4.2.2 *Modified Firefly Algorithm – Classic Bouc-Wen Model*

Table 9 contains the parameters used for the Modified Firefly Algorithm approach.

Parameter	Value
N	30
G	30
d	4
L	$[0, 0.5, -0.5, -0.5]$
U	$[2, 0.99, 0.5, 0.5]$
a	1.1
b	0.3

Table 9: Parameters for the Modified Firefly Algorithm approach – Classic Bouc-Wen model

Figure 13 shows the trend of Bouc-Wen parameters A, α, β and γ using the Modified Firefly Algorithm.

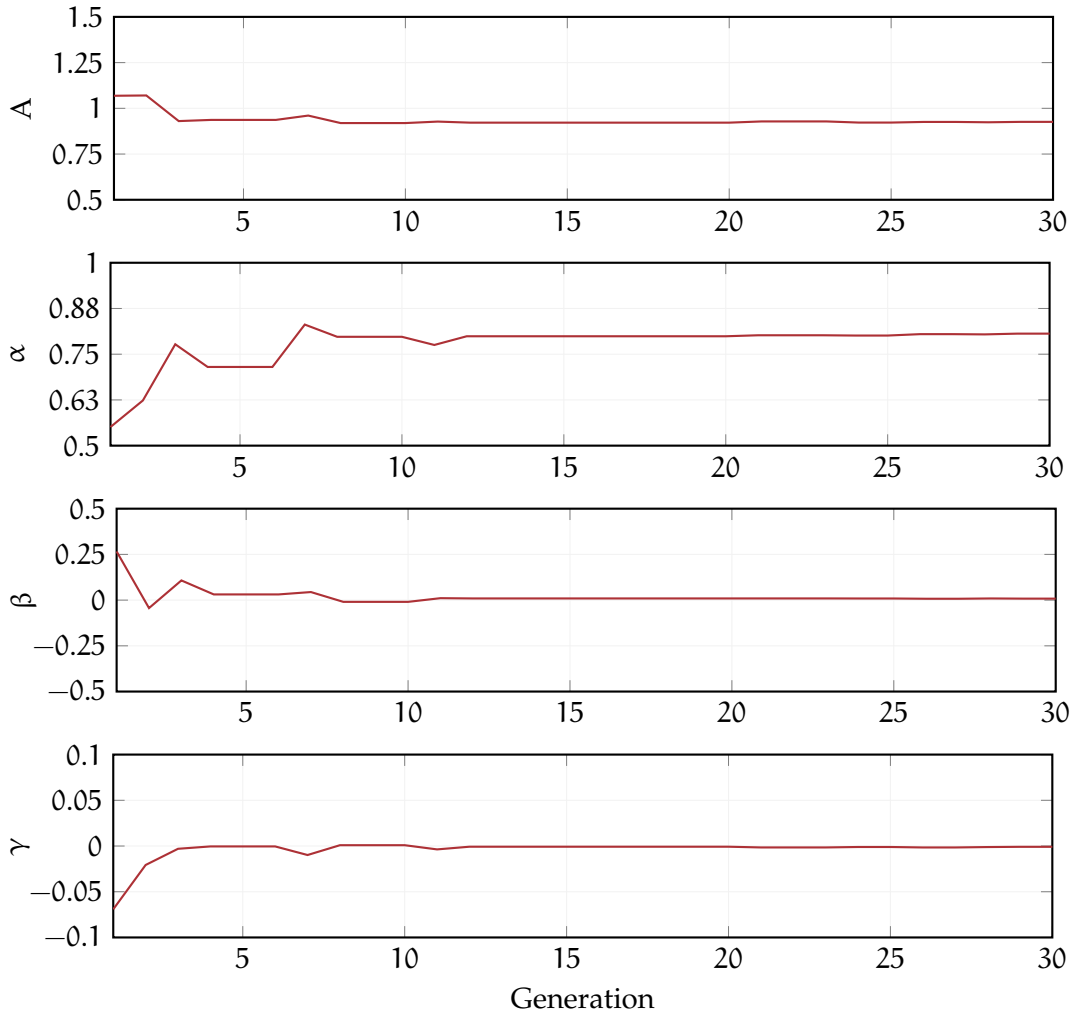

Figure 13: Modified Firefly Algorithm – Classic Bouc-Wen Model

Conversely to the Genetic Algorithm, the Modified Firefly Approach has reached good fit values in each run of the algorithm for the classic Bouc-Wen model.

Moreover, it immediately stands out how the trend of the four parameters is smoother with respect to the GA approach, for each of the three crossover strategies.

The final values found for each parameter using the MFA approach are shown in Table 10.

Parameter	A	α	β	γ
Final value	0.92596	0.80620	0.00806	−0.00086

Table 10: Modified Firefly Algorithm – Final values (Classic Bouc-Wen model)

Figure 14 shows how the hysteresis loop behaves with the final parameters of Table 10.

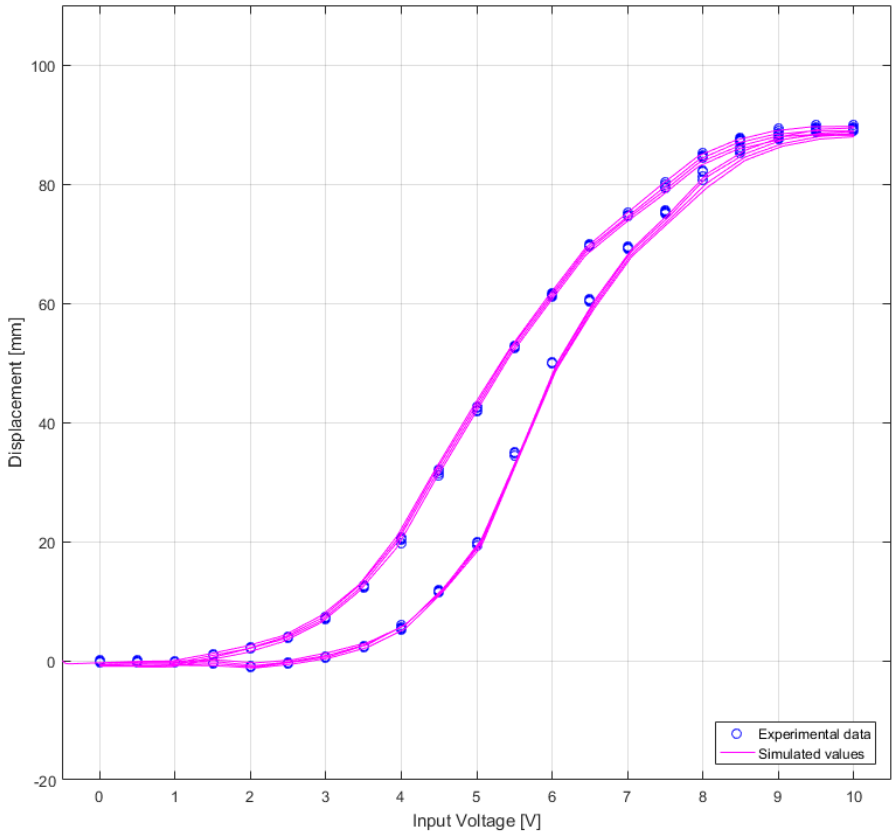

Figure 14: Hysteresis loop result using Modified Firefly Algorithm parameters (classic Bouc-Wen model)

5.4.2.3 *Particle Swarm Optimization – Classic Bouc-Wen Model*

Table 11 contains the parameters used for the Particle Swarm Optimization approach.

Parameter	Value
N	10
G	30
d	4
L	$[0, 0, -0.5, -0.5]$
U	$[2, 0.9, 0.5, 0.5]$
a	1.1
b	0.3
c_1	2
c_2	2
V_{max}	$[20, 20, 20, 20]$

Table 11: Parameters for the Particle Swarm Optimization approach – Classic Bouc-Wen model

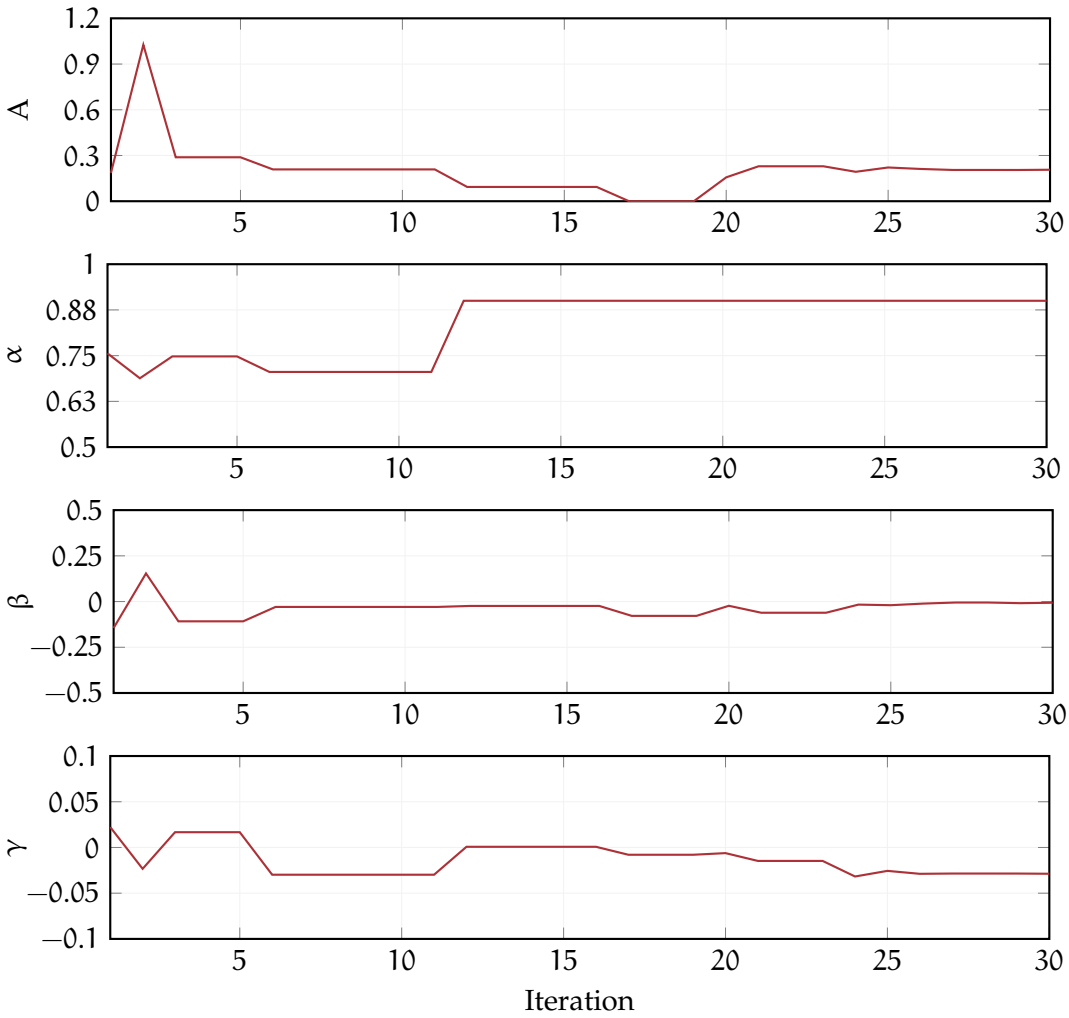

Figure 15: Particle Swarm Optimization – Classic Bouc-Wen Model

Figure 15 shows the trend of Bouc-Wen parameters using the Particle Swarm Optimization approach [3].

It is possible to notice that around iteration 15 two of the four parameters, namely α and β, have already settled and do not change much for later iterations. Thus, the particles continue to move in the solution space mainly on two coordinates. The remaining parameters A and γ stabilize over a certain value around iteration 20 and 25, respectively.

The final values found for each parameter using the PSO approach are shown in Table 12.

Parameter	A	α	β	γ
Final value	0.20651	0.9	−0.00650	−0.02874

Table 12: Particle Swarm Optimization – Final values (Classic Bouc-Wen model)

Figure 16 displays how the hysteresis loop is shaped using the final parameters of Table 12.

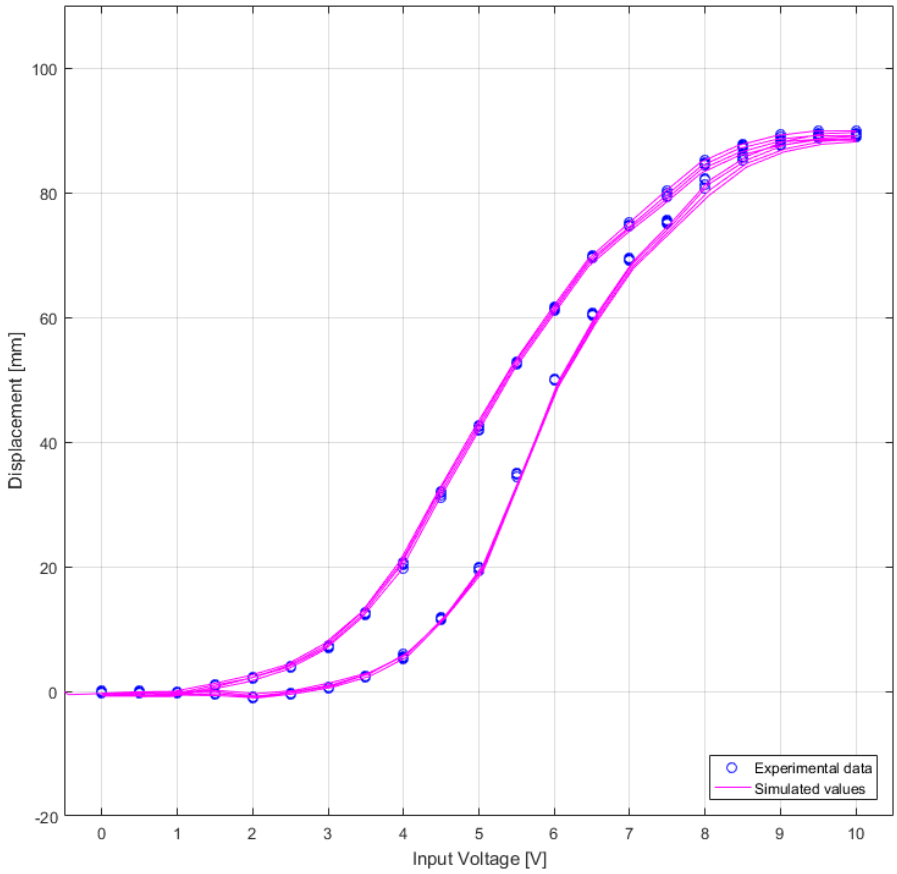

Figure 16: Hysteresis loop result using Particle Swarm Optimization parameters

3 The values shown in Figure 15 refer to the particle that attained the best global fit value (gbest) up until that iteration.

5.4.3 *Solution Comparison – Generalized Bouc-Wen Model*

This Section serves the purpose of comparing the solutions found using the algorithms described in Sections 5.2.1, 5.3.2 and 5.3.3 in order to find suitable parameters for the generalized Bouc-Wen model of hysteresis.

5.4.3.1 *Genetic Algorithm – Generalized Bouc-Wen Model*

Table 13 contains the parameters used for all three crossover strategies of the Genetic Algorithm approach.

Figures 17, 18 and 19 display the trend of the generalized Bouc-Wen model parameters A, α, β_1, β_2, β_3, β_4, β_5 and β_6 depending on the crossover method used [4].

The bounds of parameters $\beta_1, \ldots, \beta_6$ is narrower than the bounds used for the classic version of the model. This is due to the fact that this model is more complex, and thus it is harder to achieve a suitable solution.

It has been observed that the wrong choice of parameters may lead to the *explosion* of the output, i.e. computational errors add up to a point where the simulated output diverges. The BIBO stability of the Bouc-Wen model has been studied and formulated in [24].

Parameter	**Value**
N	20
G	30
d	8
L	$[0, 0.5, -0.1, -0.1, -0.1, -0.1, -0.1, -0.1]$
U	$[2, 0.99, 0.5, 0.5, 0.5, 0.5, 0.5, 1]$
P_{cross}	0.5
$Pmut$	0.1
m_m	0.05

Table 13: Parameters for the Genetic Algorithm approach – Generalized Bouc-Wen model

4 As for the classic model, Figure 17 shows, for each generation, the parameters of the best individual. This will be repeated for subsequent Figures.

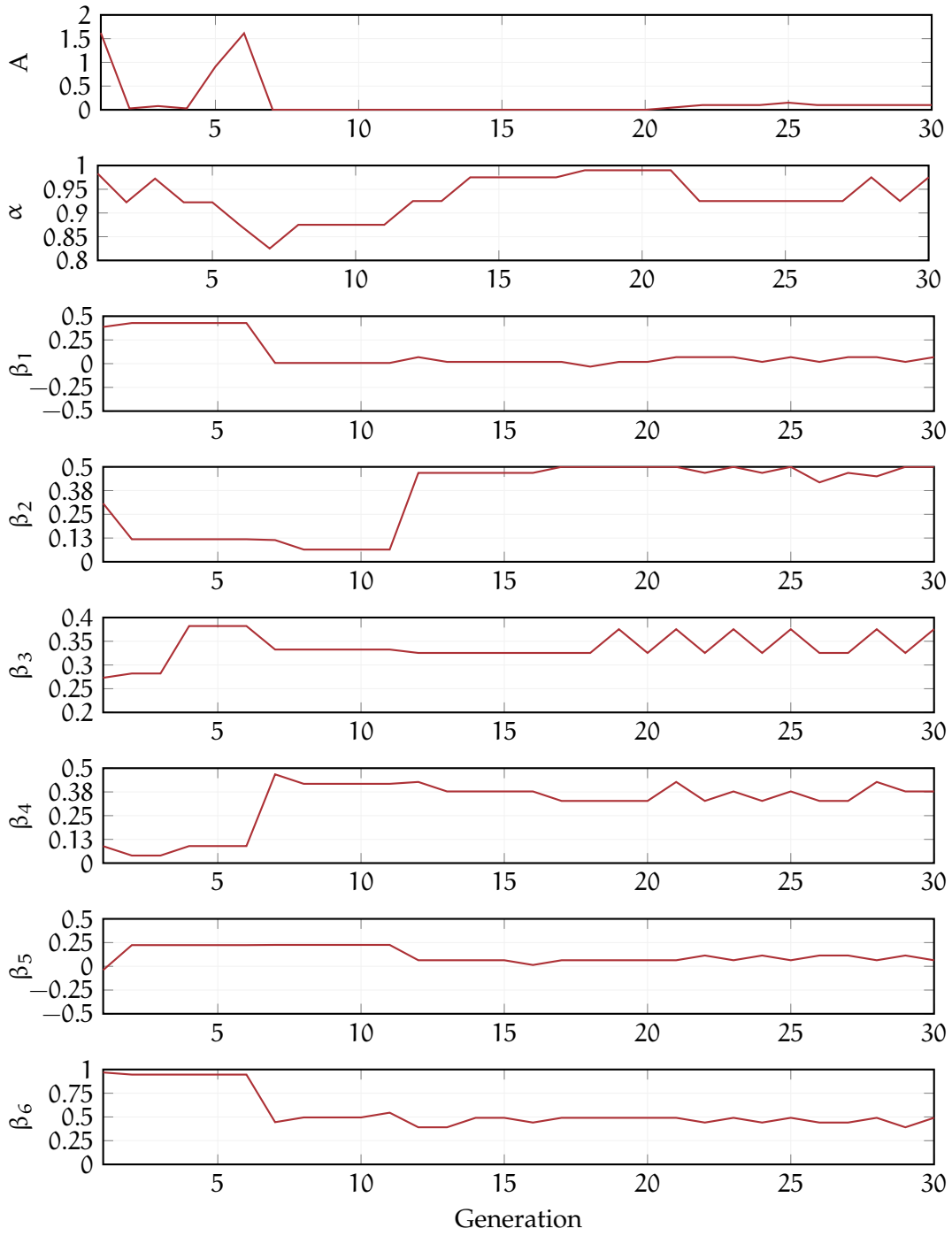

Figure 17: Genetic Algorithm – Generalized Bouc-Wen Model (Random crossover)

The final values found for each parameter using the Genetic Algorithm approach with the random crossover strategy are displayed in Table 14.

Parameter	A	α	β_1	β_2	β_3	β_4	β_5	β_6
Final value	0.1	0.9750	0.0685	0.5	0.3753	0.3775	0.0640	0.4909

Table 14: Random Crossover – Final values (Generalized Bouc-Wen model)

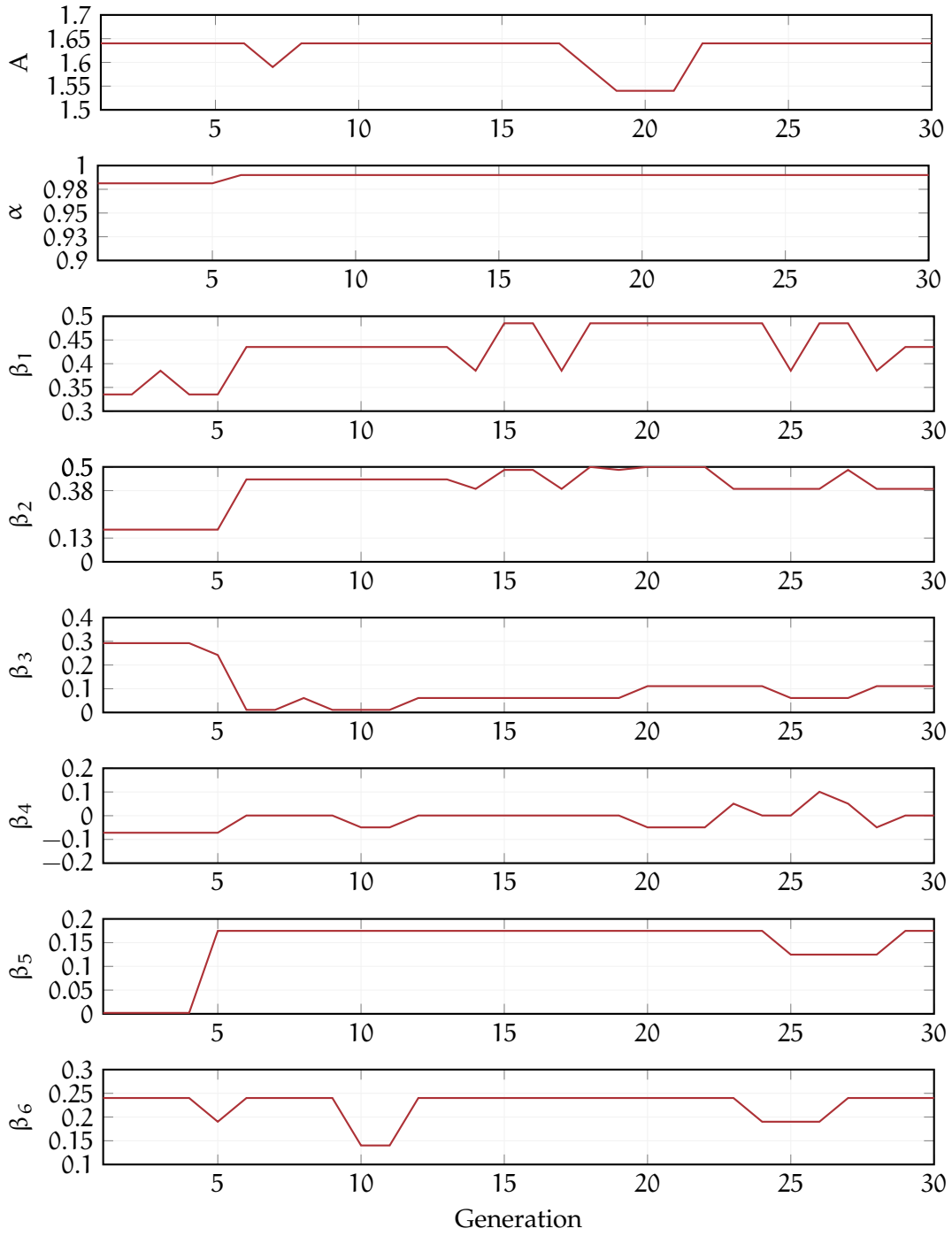

Figure 18: Genetic Algorithm – Generalized Bouc-Wen Model (Weighted crossover)

The final values found for each parameter using the Genetic Algorithm approach with the weighted crossover strategy are displayed in Table 15.

Parameter	A	α	β_1	β_2	β_3	β_4	β_5	β_6
Final value	1.6401	0.99	0.4352	0.3842	0.1110	0.0007	0.1748	0.2401

Table 15: Weighted Crossover – Final values (Generalized Bouc-Wen model)

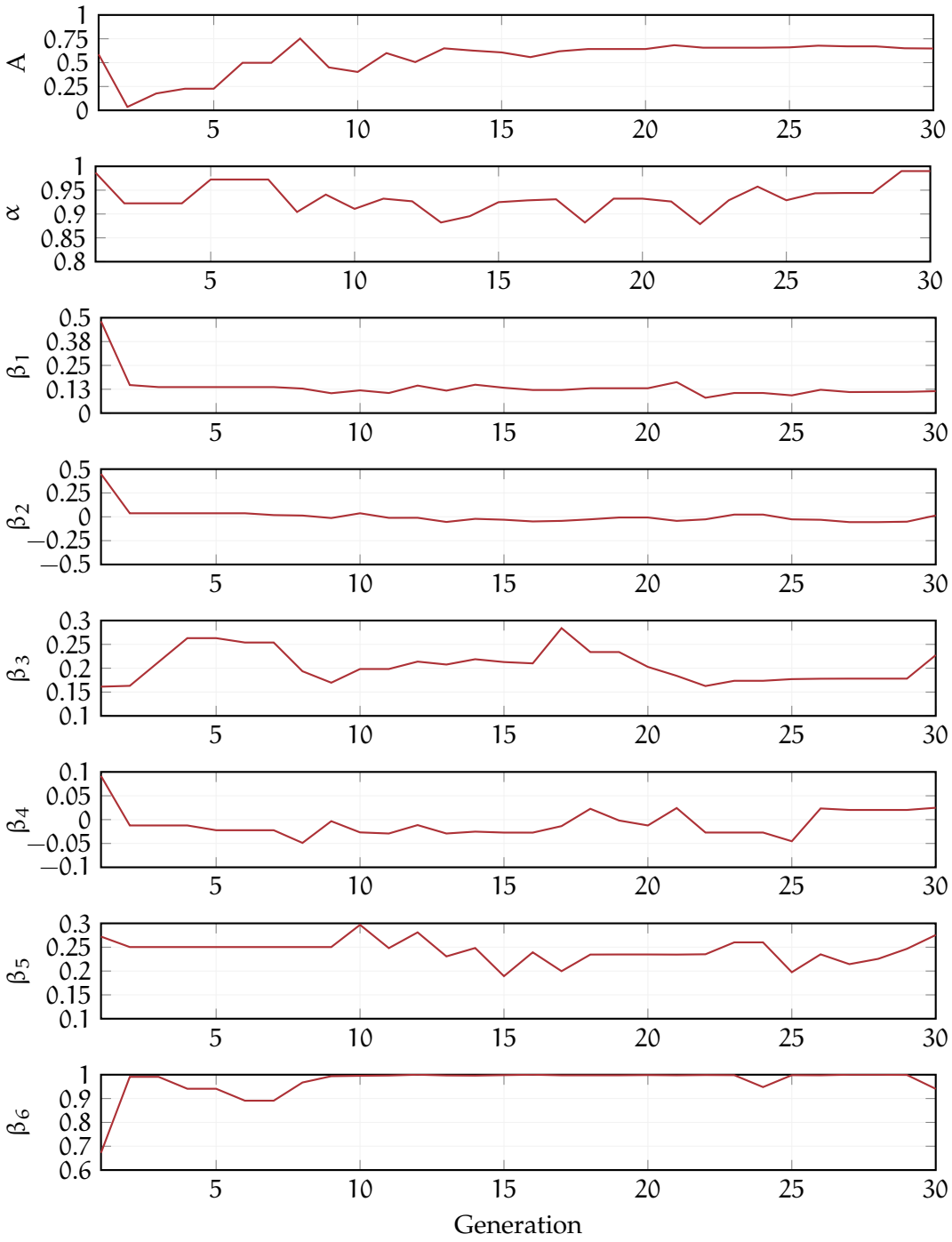

Figure 19: Genetic Algorithm – Generalized Bouc-Wen Model (Proportional crossover)

The final values found for each parameter using the Genetic Algorithm approach with the proportional crossover strategy are displayed in Table 16.

Parameter	A	α	β_1	β_2	β_3	β_4	β_5	β_6
Final value	0.6484	0.99	0.1149	0.0139	0.2274	0.0248	0.2755	0.9405

Table 16: Proportional Crossover – Final values (Generalized Bouc-Wen model)

As for the classic version of the Bouc-Wen model, it is possible to observe that for the random and weighted crossover strategies the individuals' chromosomes tend to sharply change between one value and another, as the set of possible values is that of the first generation [5]. In fact, the proportional crossover strategy figures a smoother trend for all variables.

Figure 20 shows how the hysteresis loop behaves using the final parameters of Table 16. With respect to Figure 12, a higher precision is appreciable thanks to the usage of the generalized hysteresis model.

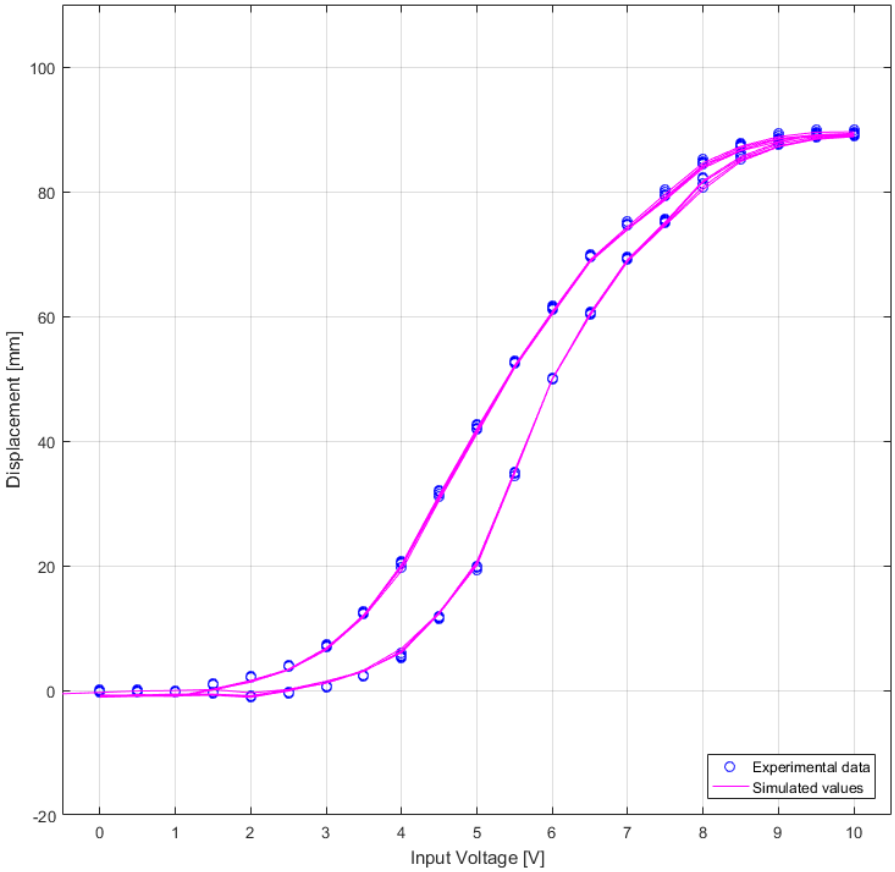

Figure 20: Hysteresis loop result – Genetic Algorithm's parameters (generalized Bouc-Wen model, proportional crossover approach)

5.4.3.2 *Modified Firefly Algorithm – Generalized Bouc-Wen Model*

Table 17 contains the parameters used for the Modified Firefly Algorithm approach.

5 Apart from eventual changes introduced by the mutation step at each generation.

Parameter	Value
N	10
G	30
d	8
L	$[0, 0.7, -0.1, -0.1, -0.1, -0.1, -0.1, -0.1]$
U	$[1.5, 1, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]$
α	1.1
b	0.3

Table 17: Parameters for the Modified Firefly Algorithm approach – Generalized Bouc-Wen model

Figure 22 shows the trend of generalized Bouc-Wen model parameters using the Modified Firefly Algorithm. The value of the best firefly for the specific generation is reported as the value outputted by the algorithm for that generation.

It is also noticeable how the values initially tend to vary a lot, as the fireflies broadly explore the solution space during the first generations, and then proceed to refine the search around the fireflies with highest lightness factor (local exploitation).

The hysteresis loop behaviour with the final parameters is displayed in Figure 21.

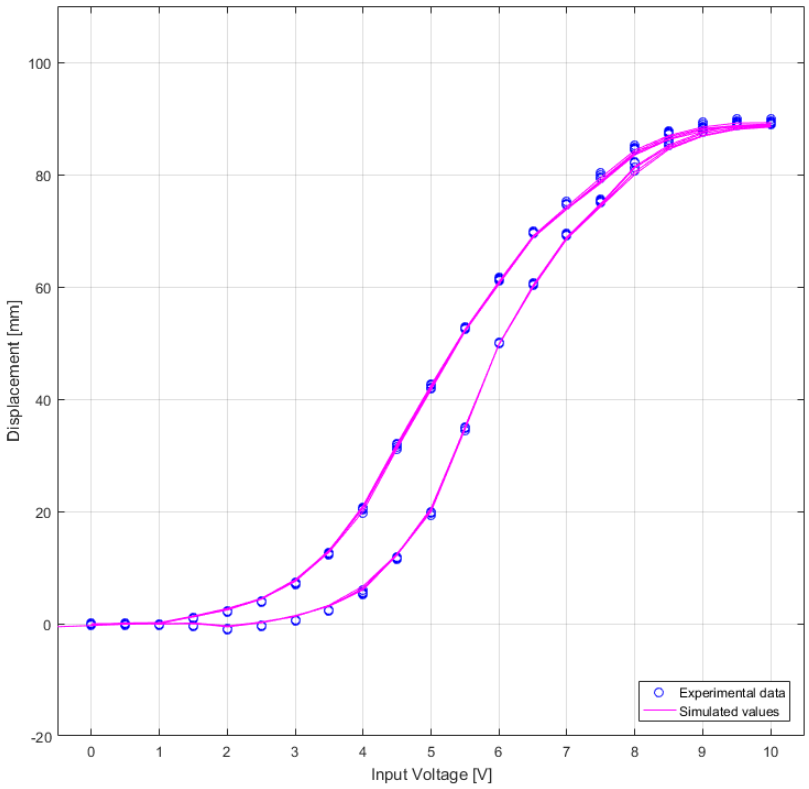

Figure 21: Hysteresis loop result using Modified Firefly's parameters (generalized Bouc-Wen model)

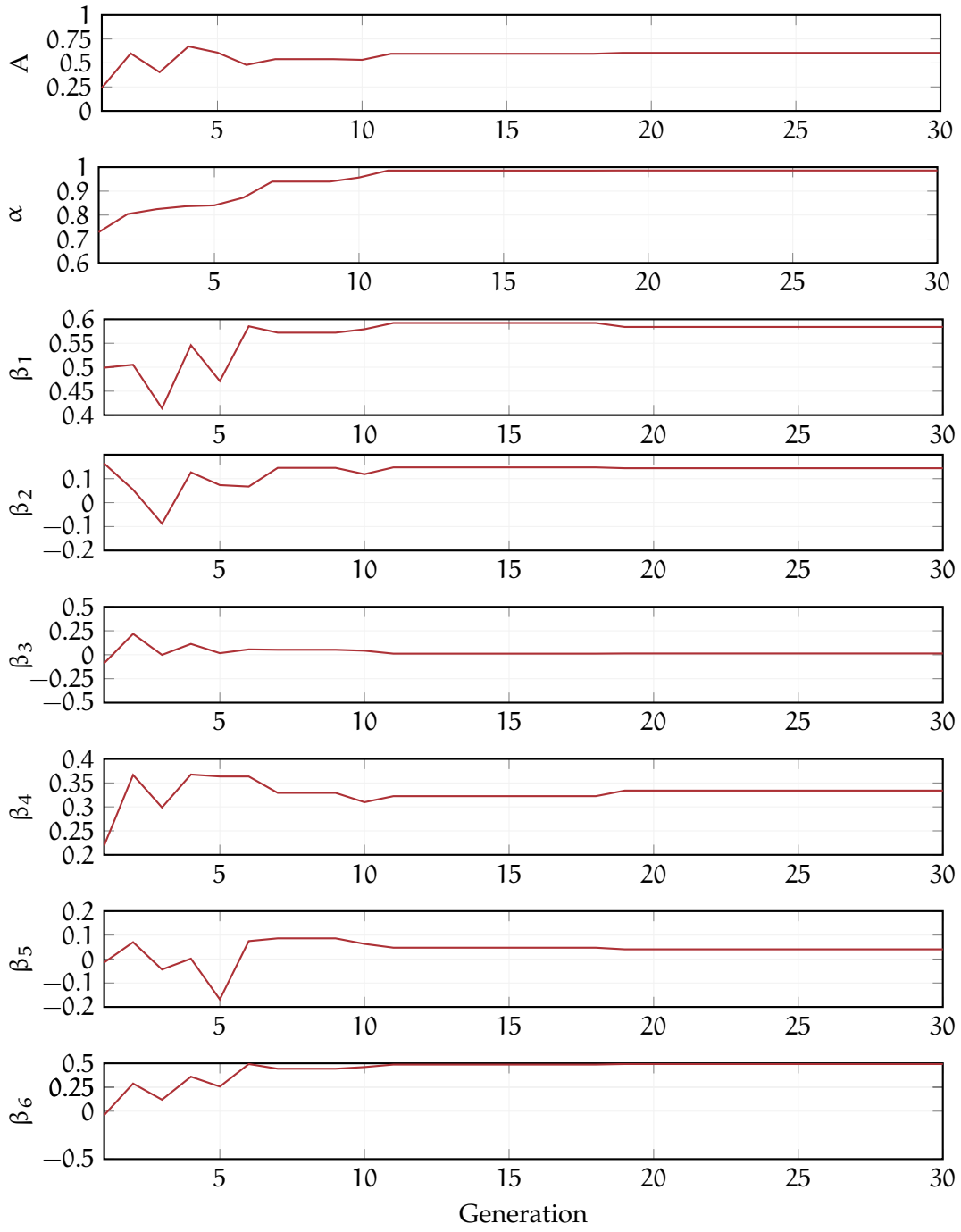

Figure 22: Modified Firefly Algorithm – Generalized Bouc-Wen Model

The final values found for each parameter using the Modified Firefly Algorithm approach are displayed in Table 18.

Parameter	A	α	β_1	β_2	β_3	β_4	β_5	β_6
Final value	0.6049	0.9861	0.5840	0.1435	0.0134	0.3341	0.0405	0.4924

Table 18: Modified Firefly Algorithm – Final values (Generalized Bouc-Wen model)

5.4.3.3 *Particle Swarm Optimization – Generalized Bouc-Wen Model*

Table 19 contains the parameters used for the Particle Swarm Optimization approach. The hysteresis loop behaviour with the final parameters is displayed in Figure 23. Figure 24 shows the trend of generalized Bouc-Wen model parameters using the Particle Swarm Optimization approach.

Parameter	**Value**
N	10
G	30
d	8
L	$[0, 0, -0.1, -0.1, -0.1, -0.1, -0.1, -0.1]$
U	$[2, 0.9, 1, 1, 1, 1, 1, 1]$
a	2.1
b	0.6
c_1	2
c_2	2
V_{max}	$[0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1]$

Table 19: Parameters for the Particle Swarm Optimization approach – Generalized Bouc-Wen model

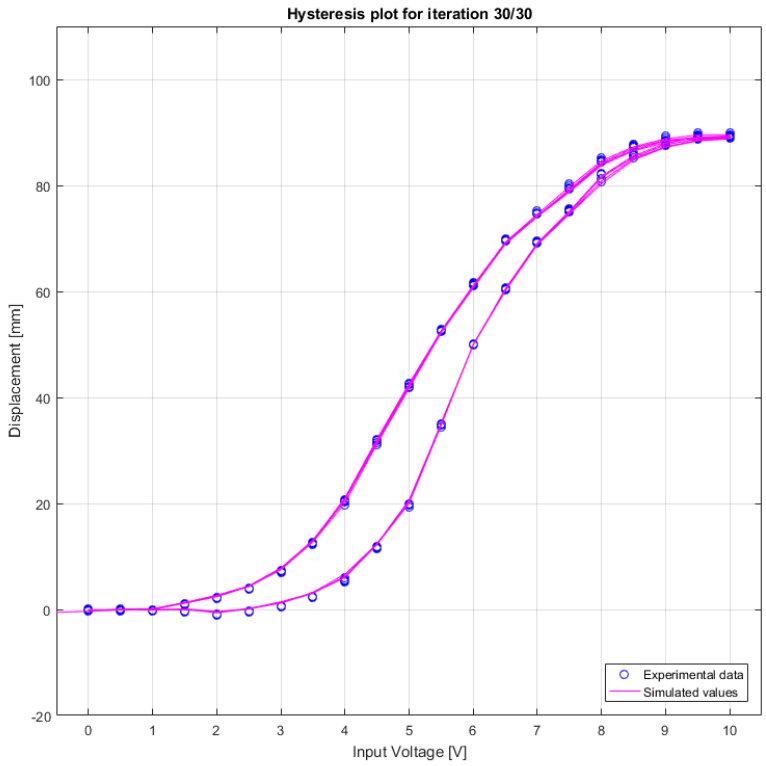

Figure 23: Hysteresis loop result using Particle Swarm Optimization parameters (generalized Bouc-Wen model)

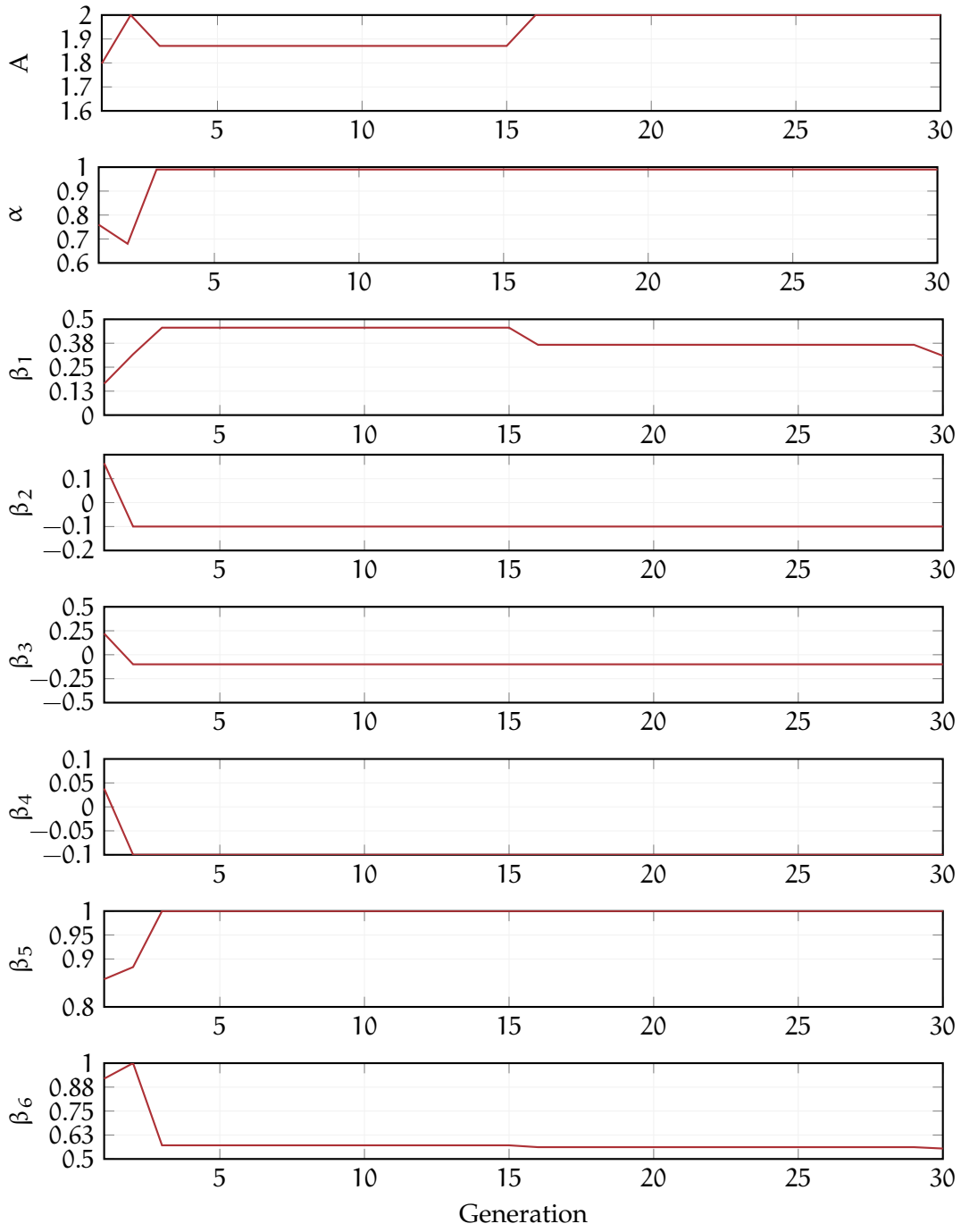

Figure 24: Particle Swarm Optimization – Generalized Bouc-Wen Model

The final values found for each parameter using the Particle Swarm Optimization approach are displayed in Table 20.

Parameter	A	α	β_1	β_2	β_3	β_4	β_5	β_6
Final value	2	0.99	0.3096	−0.1	−0.1	−0.1	1	0.5545

Table 20: Particle Swarm Optimization – Final values (Generalized Bouc-Wen model)

5.4.4 *Time Comparison*

In order to be relevant, timing comparisons among the algorithms previously described have to be made with the same number of individuals in the population. This ensures impartiality between them when it comes to ranking the algorithms for performance.

Each algorithm has been thus ran with a variable number of candidate solutions of 10, 20 and 30, and the total execution time has been recorded. For each algorithm and each number of candidate solutions, a average execution time per generation is presented. The intent is to rank them by execution time and relative precision.

Table 21 displays the differences in execution times for all algorithms with different numbers of individuals in the population. The number of generations/iterations has been set to 30 for all algorithms.

For the Genetic Algorithm approach, it has been observed that the crossover strategy chosen does not affect the execution time. For this reason the values presented in the table refer to execution times obtained with the proportional crossover method.

Number of individuals	Algorithm	Total time (Classic)	Average Time (Classic)	Total Time (Generalized)	Average Time (Generalized)
10	GA	21.98 s	0.73 s	30.72 s	1.02 s
	MFA	21.34 s	0.71 s	30.78 s	1.03 s
	PSO	21.36 s	0.71 s	30.57 s	1.02 s
20	GA	37.96 s	1.26 s	59.65 s	1.99 s
	MFA	39.43 s	1.31 s	60.36 s	2.01 s
	PSO	40.84 s	1.36 s	62.79 s	2.09 s
30	GA	55.45 s	1.85 s	84.18 s	2.81 s
	MFA	54.32 s	1.81 s	85.04 s	2.83 s
	PSO	55.40 s	1.84 s	82.31 s	2.74 s

Table 21: Timing performance comparison among algorithms

Timing results of Table 21 must be compared also to the relative precision of the final solution proposed by each algorithm: as a matter of fact, in the case of the Genetic Algorithm method, a lower number of individuals in the population brings timing results almost identical to those of the other two algorithms, but the precision heavily depend on the chromosomes of individuals that are generated initially. In the other two algorithms, the global exploration approach undertaken by the fireflies/particles make them wander faster across the solution space. Therefore, the lower number of individuals does not cause a lower precision as much.

The timing performance of the three algorithms make them not suitable for online control. There may be an improvement in performance if the algorithms were implemented onto dedicated hardware, e.g. a single purpose processor or a video card programmed to faster calculate the evolution of individuals, or the movement of fireflies/particles.

6 CONTROLLER DESIGN

6.1 INTRODUCTION

To achieve precise position control, a PI controller has been developed, simulated and then tested onto real equipment. The controller has been tested using a sine wave reference with different frequencies for each run, from 1 to 3 $^{rad}/_{s}$.

The weight lifted by the muscle is also variable, and experiments have been done with weights of 4.5 kg and 7 kg. To connect the MATLAB suite and the real apparatus, a Simulink model has been developed. The connection is achieved by using a dSPACE [25] DS1104 Controller Board, which upgrades a PC to a development system for rapid control prototyping, and that can be seen in Figure 25. The Controller Board can be installed in any PCI or PCIe slot.

The Controller Board has been connected to several components, to monitor important model variables (i.e. the muscle's pressure) and apply control input to two proportional valves.

Figure 25: DS1104 R&D Controller Board

6.2 PI CONTROLLER DEVELOPMENT

6.2.1 *Introduction on PID Control*

Figure 26 shows a block diagram of a PID controller in a feedback loop. The idea behind PID control is to keep track of the error and trying to minimize it, using three control terms of proportional, integral and derivative. The action of these three control terms influence the controller output in order to apply accurate and optimal control.

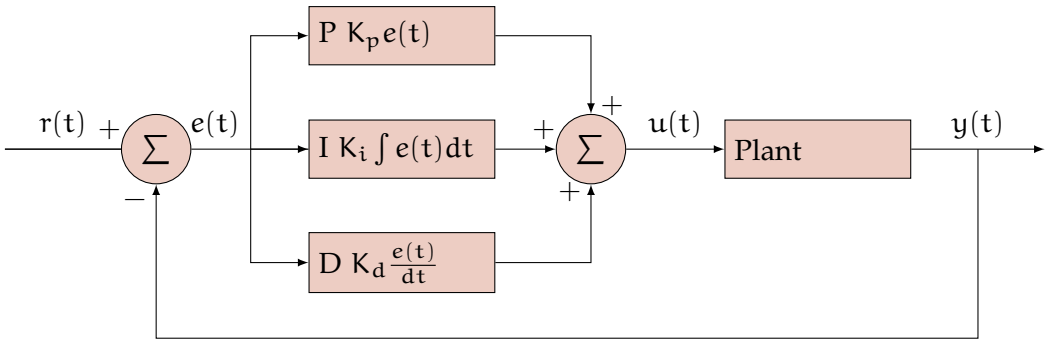

Figure 26: Block diagram of a PID controller in a feedback loop

Given the error $e(t)$ as the difference between the reference $r(t)$ and the output $y(t)$, the three components act as follows:

- The term P is proportional to the current value of $e(t)$.
- The term I takes into account past values of $e(t)$. The integral term seeks to eliminate the residual error by adding a control effect due to the historic cumulative value of the error.
- The term D is a best estimate of the future trend of $e(t)$. The more rapid the error changes, the greater will be the controlling or dampening effect

The overall control function can be expressed mathematically as in Equation 41.

$$u(t) = K_p e(t) + K_i \int_0^t e(t)dt + K_d \frac{de(t)}{dt} \tag{41}$$

The PID control is easy to develop and to apply. The main drawback is the tuning of the control gains K_i, K_i and K_p depending on the features of the system. For example, in the case of McKibben muscles it may be necessary to tune the control when the load is changed or for different kinds of muscles.

The use of the PID algorithm does not guarantee optimal control of the system nor its control stability. Moreover situations may arise where there are excessive delays: the measurement of the process value is delayed, or the control action does not apply quickly enough to guarantee stability.

6.2.2 *Application of PI Control*

As introduced in Section 6.1, the control has been applied onto a real McKibben muscle in a laboratory environment. The reference $r(t)$ has been set as a sine wave, with variable frequency of 1, 2 and 3 $^{rad}/_{s}$.

Section 6.2.2.1 covers the application of PI control on the laboratory muscle with a weight of 4.5 kilograms, and Section 6.2.2.2 concerns the application of PI control with a weight of 7 kilograms.

The hysteresis model used for control is the classic Bouc-Wen model. The model parameters used are obtained offline beforehand using the Modified Firefly Algorithm, described in Section 5.3.2.

The reference signal $r(t)$ has an amplitude of 10 millimetres, a bias of 30 and variable frequency. The code to generate this kind of wave is in Appendix A.

6.2.2.1 *Application on Muscle – 4.5 Kilograms Weight*

This Section covers the application of PI control on a real McKibben artificial muscle driven by tap-water. The weight attached to the muscle weighs 4.5 kilograms.

The PID gains have been set as those of Table 22 for all three frequencies, to show the limitations of this kind of control.

Gain parameter	K_p	K_i	K_d
Value	0.06875	0.05	0

Table 22: PID gains for 4.5 kilograms

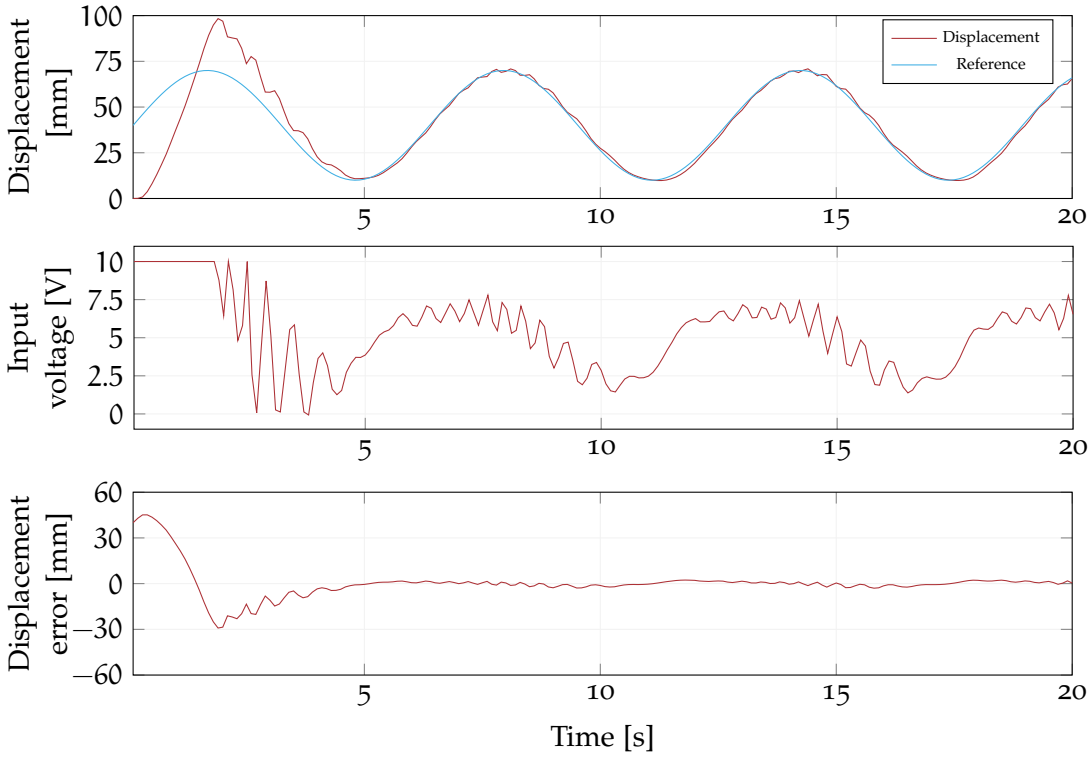

Figure 27: Displacement control, 4.5 kilograms, 1 rad/s

For each Figure, the first plot shows a comparison between the reference signal and the effective displacement of the muscle. The second plot shows the voltage applied to the input proportional valve. Lastly, the third plot shows the displacement error in millimetres, that is the difference between the reference value and the muscle displacement.

As it is possible to see in Figure 27 and 28, the reference tracking is very precise for 1 and 2 rad/s.

On the other hand, it is possible to notice a steady degradation in performance as the wave frequency rises. This is especially noticeable in Figure 29, for a sine wave with a frequency of 3 rad/s, where a significant delay of approximately 1 second can be appreciated between the reference and the effective displacement.

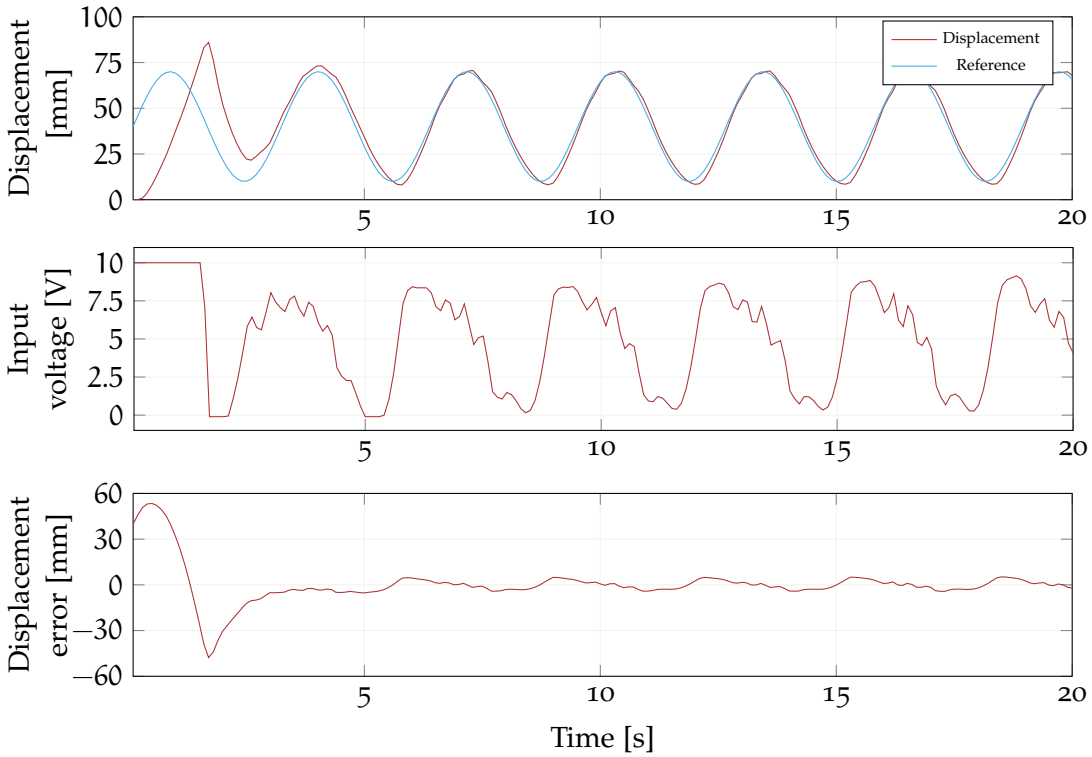

Figure 28: Displacement control, 4.5 kilograms, 2 $^{rad}/_{s}$

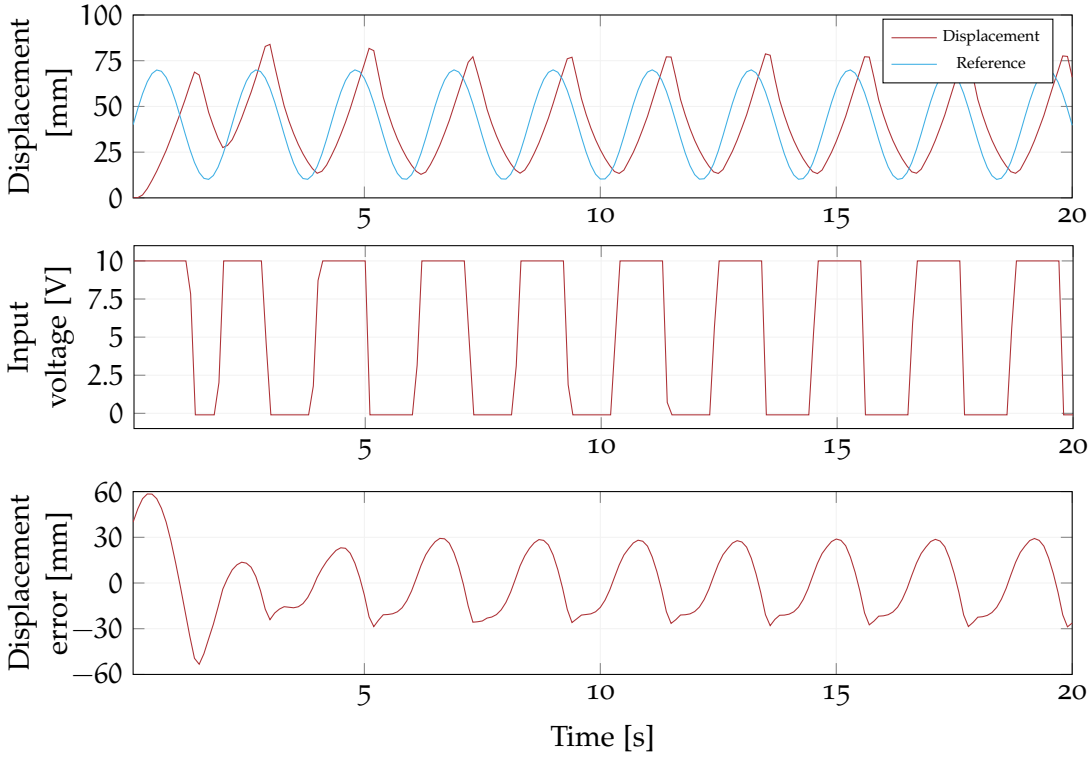

Figure 29: Displacement control, 4.5 kilograms, 3 $^{rad}/_{s}$

6.2.2.2 *Application on Muscle – 7 Kilograms Weight*

This Section covers the application of PI control on a real McKibben artificial muscle driven by tap-water. The weight attached to the muscle weighs 4.5 kilograms.

The PID gains have been set as those of Table 22 for all three frequencies, to study the effect of the weight on control performances.

The main difference between the control performance with a weight of 4.5 kilograms is the case with a reference with frequency of 3 $^{rad}/_{s}$. Although there is still a relevant error due to the delay in following the reference, it has lower impact on the control performance, meaning that the displacement error is lower. This is probably due to the greater inertia caused by the heavier weight. It is possible to appreciate these differences in Figure 32.

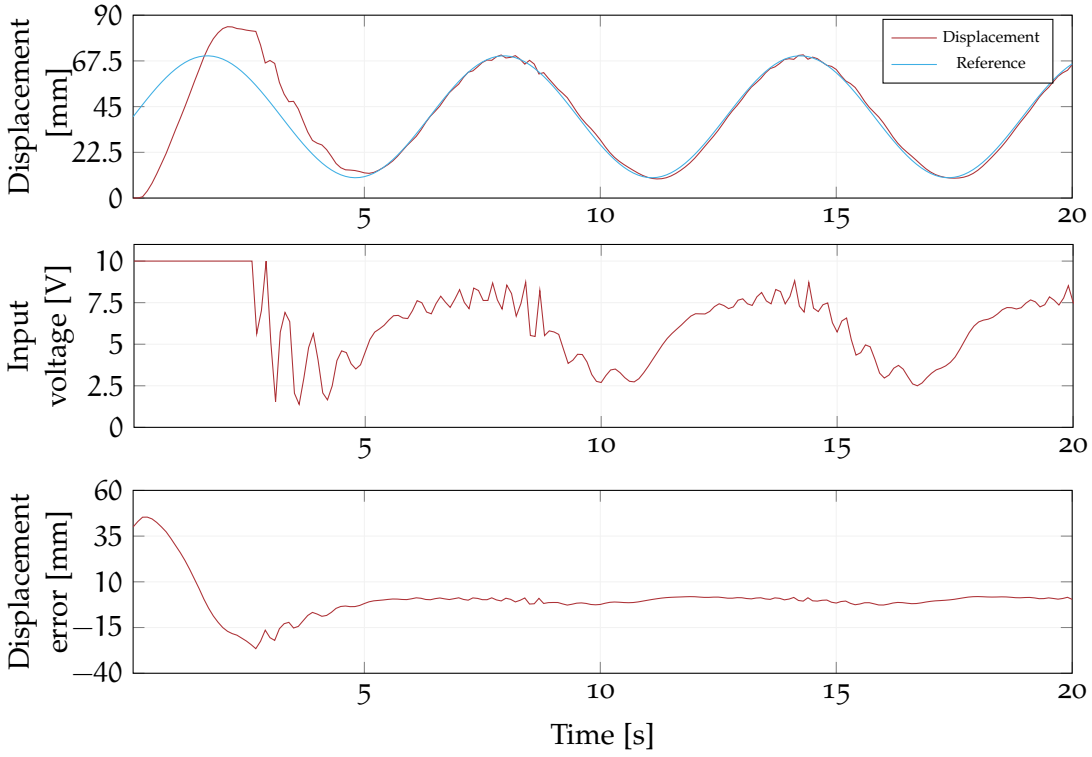

Figure 30: Displacement control, 7 kilograms, 1 $^{rad}/_{s}$

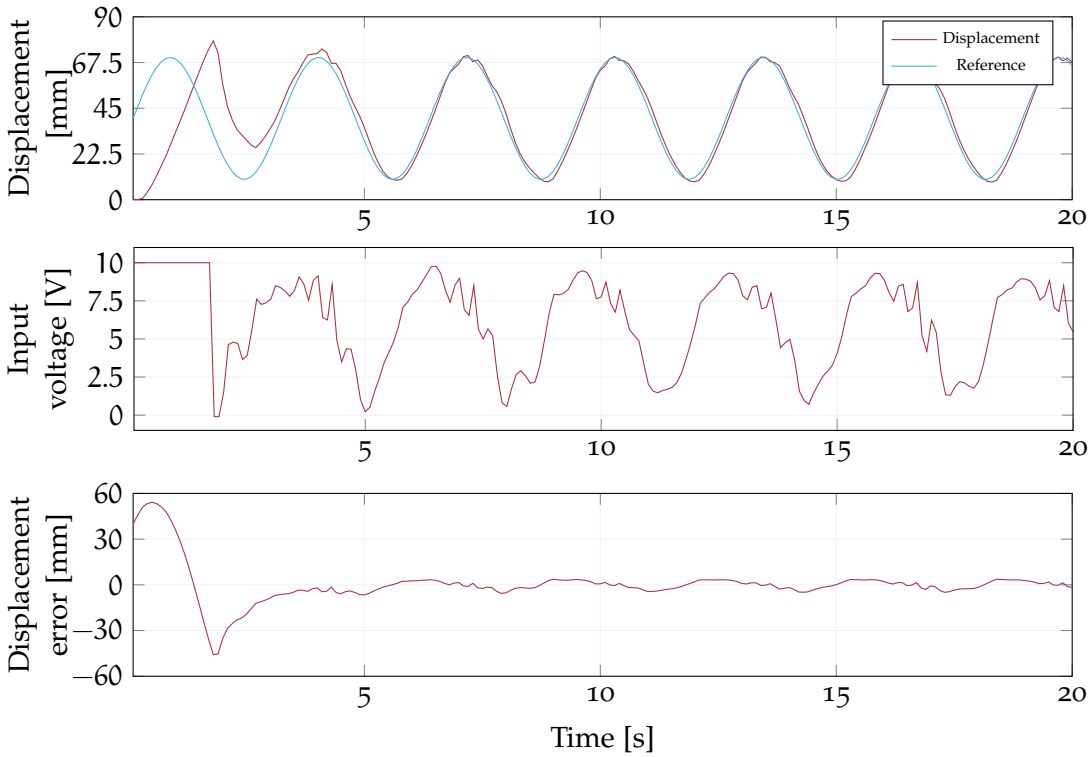

Figure 31: Displacement control, 7 kilograms, 2 $^{rad}/_{s}$

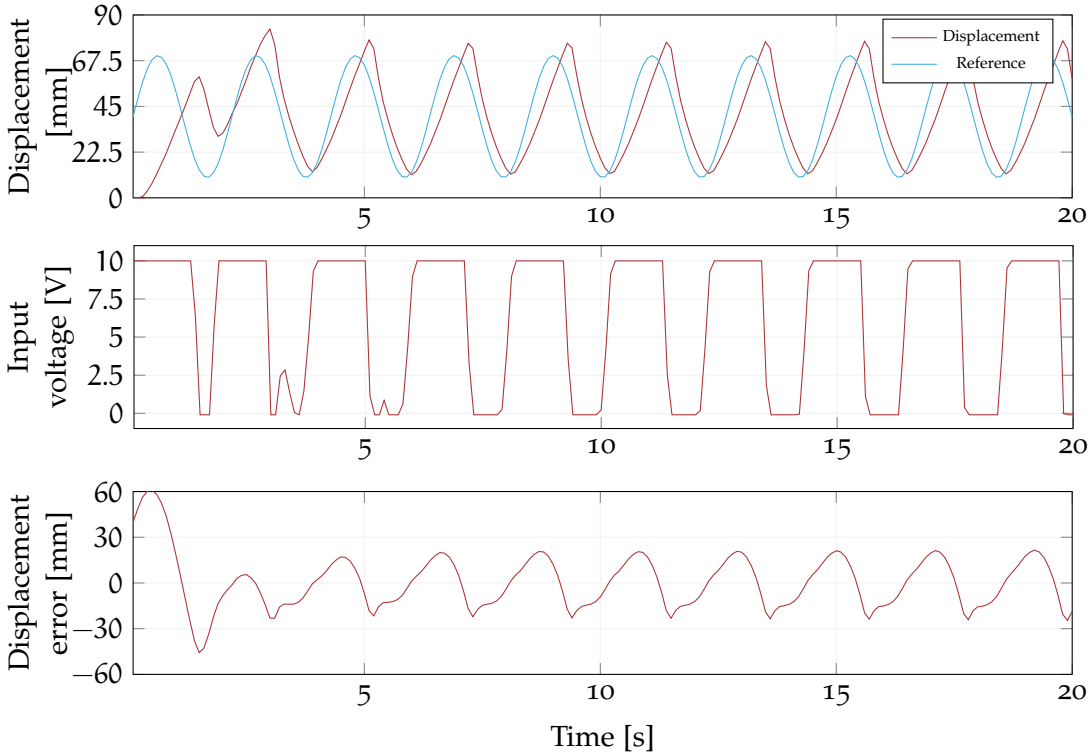

Figure 32: Displacement control, 7 kilograms, 3 $^{rad}/_{s}$

6.3 MODEL PREDICTIVE CONTROL DEVELOPMENT

A MPC approach has been simulated using an online tool for convex optimization called CVXGEN [26]. This tool allows to write a linear programming problem description and generate a solver that can be embedded in a Simulink block for fast and efficient computation. A problem description example for a MPC is in Figure 33.

```
dimensions
  m = 1   # inputs.
  n = 3   # states.
  T = 10  # horizon.
end

parameters
  A (n,n)      # dynamics matrix.
  B (n,m)      # transfer matrix.
  Q (n,n) psd  # state cost.
  Q_final (n,n) psd  # final state cost.
  R (m,m) psd  # input cost.
  x[0] (n)     # initial state.
  u_max nonnegative  # amplitude limit.
  S nonnegative      # slew rate limit.
end

variables
  x[t] (n), t=1..T+1  # state.
  u[t] (m), t=0..T    # input.
end

minimize
  sum[t=0..T](quad(x[t], Q) + quad(u[t], R)) + quad(x[T+1], Q_final)
subject to
  x[t+1] == A*x[t] + B*u[t], t=0..T  # dynamics constraints.
  abs(u[t]) <= u_max, t=0..T  # maximum input box constraint.
  norminf(u[t+1] - u[t]) <= S, t=0..T-1  # slew rate constraint.
end
```

Figure 33: MPC problem description example using CVXGEN

A prediction horizon of 8 steps has been chosen beforehand. The number of inputs is one (the voltage), and the system used for control simulation is of the third order. This has been obtained using the MATLAB System Identification Toolbox as for the transfer function described in Chapter 4.

However, it appears that the introduction of the hysteresis model renders the solver unable to solve the minimization problem introduced with the Model Predictive Control. This is probably due to the fact that the introduction of the Bouc-Wen model of hysteresis changes the problem and makes it no more convex. Because of this, the solver generated by CVXGEN is unable to solve the problem correctly.

For reasons of time, this point has been left as future work.

7 CONCLUSIONS AND FUTURE DEVELOPMENTS

McKibben muscles have proven to be of large utility in many fields, such as rehabilitation, training and actuation of small scale machinery.

The use of tap water as a medium to raise the internal pressure of this kind of muscles has many advantages: being oil free, they are naturally environment-friendly. Moreover, the use of water instead of oil makes them safer to use in specific situations.

However, there are drawbacks: water brings problems such as cavitation, and its lower viscosity makes it harder to control. Furthermore, the McKibben artificial muscle present a hysteretic behaviour that need to be accounted for to achieve good control performance.

In this dissertation, the Bouc-Wen model of hysteresis has been proposed to model the hysteresis of the muscle. This allows to get a very simple, yet precise model that can be easily controlled using techniques like PID or MPC control. The Bouc-Wen model has been proposed both in its standard form, that has been referred as *classic*, and a *generalized* form. Both versions can express a hysteretic behaviour by use of several parameters: 4 in case of the classic model, and 8 in case of the generalized one.

Finding these parameters to precisely model a given hysteresis is a hard matter. While values can be cherry-picked, to achieve highest grades of precision different methods are needed. In this work, three distinct approaches have been studied and developed: a Genetic Algorithm, that falls in the category of Evolutionary Algorithms and that mimics the natural evolution that permeates our world. Secondly, the Firefly Algorithm that mimics the behaviour of fireflies in the wild, in its modified version. Lastly, a Particle Swarm Optimization approach where candidate solutions, called particles, are flown into the solution space and evaluate it trying to find a global optimum. These last two approaches fall into the category of Metaheuristic Optimization algorithms.

By using these algorithms it has been showed that it is possible to find parameters that allow to find a good fit for the hysteresis loop. Execution times have also been studied to provide a relative rank by timing performance. This kind of algorithm may be embedded on dedicated hardware to further speed up the process. This approach may be used to find parameters for other models and other applications, for example using it to identify the best gains for a PID controller.

7.1 FUTURE WORK

Future work includes:

- Development of Model Predictive Control with the McKibben artificial muscle
- Implementation of other Evolutionary Algorithm approaches applied to muscle control

- Extension of functionality to the Genetic Algorithm approach to better model the tournament, crossover and mutation steps
- Addition of features to the Particle Swarm Optimization approach to attain higher versatility

Part II

APPENDICES

A

SCRIPTS AND ALGORITHMS

This Appendix contains all scripts and algorithms used for this work.

A.1 SCRIPTS

A.1.1 *Stair Input Generation*

This script generates the step wave that is to be given as input to the input and output proportional valves. Given that they have to be symmetric, the wave given to the output valve is defined as follows.

$$y' = 10 - y$$

Stair input generation

```
close all
clc
step = 1;       % Volts between two consecutive steps
y = zeros(1,50*10*(1/step));
for v = 0:step:10
        y((v*50)*(1/step)+1:(v+step)*(1/step)*50) = v;
end
y = [y fliplr(y)];
y = [y y y y]';
last = (length(y)-1)/10;
t = (0:0.1:last)';
y = [t y];
```

A.1.2 *Reference Signal Generation*

This script generates the reference signal used in the PID control as described in Section 6.2.2.1 and 6.2.2.2.

Reference signal generation

```
close all
clc
total_time = 20;
t = 1:0.1:20;
amplitude = 10;
bias = 30:
frequency = 1; % Can be 1, 2, 3 rad/s
ref = amplitude*sin(2*pi*frequency)+bias;
```

A.2 OPTIMIZATION ALGORITHMS

This Section contains the MATLAB code of the Genetic Algorithm, the Modified Firefly Algorithm and the Particle Swarm Optimization. For presentation purposes, some of the code has been split into separate functions.

A.2.1 *Genetic Algorithm*

A.2.1.1 *Algorithm Initialization*

In the initialization step, all variables and parameters of all the algorithm are initialized. This step is also performed for the Modified Firefly Algorithm and the Particle Swarm Optimization approach.

Genetic Algorithm initialization

```
% Population parameters
n = 30;                    % Initial population size
generations = 30;          % Number of generations
d = 4;                     % Chromosome dimension
Lb = [0.5 0.5 -0.5 -0.1]; % Parameters' lower bound
Ub = [1 0.99 0.5 0.1];     % Parameters' upper bound
% Genetic parameters
crossover_mode = "PROPORTIONAL";
p_crossover = 0.5;           % Probability of crossover
p_mutation = 0.1;            % Probability of mutation
mutation_magnitude = 0.05;  % Percentage of mutation
% Population initialization
pop = zeros(n,d);            % Population matrix
mating_pool = zeros(n,d);   % Mating pool
mating_pool_dim = 0;
fit = zeros(n,1);            % Fit vector
relative_fit = zeros(n,1);
```

A.2.1.2 *Population Evaluation*

Population evaluation

```
for k=1:generations
  best=zeros(1,generations);
  for i=1:n
  % Individual evaluation (code removed)
  [response,individual] = checkParameters(pop(i,:),Lb,Ub);
  if response % Checking if values are inside the bounds
    fit(i) = -1/sum((eval_points_out-eval_points_ref).^2);
  else
    pop(i,:) = individual; % Keep the old individual
    fit(i) = -1/(sum((eval_points_out-eval_points_ref).^2)^5);
  end
  matingStep();
  mutationStep();
end
```

A.2.1.3 *Parameter Checking for New Individuals*

Parameter checking

```
function [response,individual] = checkParameters(individual,Lb,Ub)
  response = true;
  for i=1:length(Lb)
    if individual(i)<Lb(i)
      individual(i)=Lb(i);
      response=false;
    elseif individual(i)>Ub(i)
      individual(i)=Ub(i);
      response=false;
    end
  end
end
```

A.2.1.4 *Mating Step*

Mating step overview

```
function mating_pool = matingStep(mating_pool,mating_pool_dim,pop,fit,d,
    crossover_mode,cross_chance,p_crossover)
  while mating_pool_dim < n
    % Select individuals to mate until the mating pool is full
    % The crossover process depends on the fitness value of the parents
    [first,second] = getMatingPartners(pop,fit);
    cross_chance = fit(first)/(fit(first)+fit(second));
    [first_child,second_child] = getCrossoverFromParents(pop(first,:),
        pop(second,:),d,crossover_mode,cross_chance,p_crossover);
    mating_pool(mating_pool_dim+1,:) = first_child;
    mating_pool(mating_pool_dim+2,:) = second_child;
    mating_pool_dim = mating_pool_dim+2;
  end
end
```

A.2.1.5 *Mating Partners Selection*

Mating partners selection

```
function [first,second] = getMatingPartners(pop,fit)
  fit_copy = fit/sum(fit);
  cumulative = 0;
  index = 1;
  while cumulative <= rand
    cumulative = cumulative + fit_copy(index);
    index = index + 1;
  end
  first = index;
  % Repeating for second parent
  fit_copy = fit;
  fit_copy(index,:) = [];
  fit_copy = fit_copy/sum(fit_copy);
  index = 1;
  cumulative = 0;
  while cumulative <= rand
    cumulative = cumulative + fit_copy(index);
    index = index + 1;
  end
  second = index;
  if(first == second)
    second = second + 1;
  end
end
```

A.2.1.6 *Crossover*

Mating partners selection

```
function [first_child,second_child] = getCrossoverFromParents(first,
    second,d,method,fit_first,p_crossover)
  first_child = first;
  second_child = second;
  % Random and weighted crossover strategies omitted for presentation
  if strcmp(method,"PROPORTIONAL")
    for x=1:d
      if rand>p_crossover
        first_child(x)=fit_first*first(x)+(1-fit_first)*second(x);
      end
    end
    for x=1:d
      if rand>p_crossover
        second_child(x)=fit_first*first(x)+(1-fit_first)*second(x);
      end
    end
  end
end
```

A.2.2 *Modified Firefly Algorithm*

Firefly evaluation and movement

```
for k = 1:generations
  [lambda_0, sigma, rho] = update_params(k,a,b,generations);
  % Firefly evaluation omitted
  % Firefly movement
  for i=1:n
    ff_i = ns(i,:);
    ill_i = fit(i);
    for j=1:n
      if fit(j) > ill_i
        r=sqrt(sum((ns(i,:)-ns(j,:)).^2));
        lambda_ij = lambda_0 * exp(-sigma*r);
        ns(i,:) = ns(i,:)+lambda_ij*(ns(j,:)-ns(i,:))+rho*(randn(1,d));
        for par = 1:d
          if ns(i,par)<Lb(par)
            ns(i,par)=Lb(par);
          elseif ns(i,par)>Ub(par)
            ns(i,par)=Ub(par);
          end
        end
      end
    end
  end
end
```

A.2.2.1 *Firefly Algorithm Parameter Update*

Parameter update

```
function [lambda_0,sigma,rho]=update_params(k,a,b,generations)
  lambda_0 = a-(a-b)*(1-k)/(1-generations);
  sigma = 2*7^(-k/generations);
  rho = 0.1*exp(-4.8*k/generations);
end
```

A.2.3 *Particle Swarm Optimization*

A.2.3.1 *Evaluation and Movement*

Particle Swarm Optimization algorithm

```
for k=1:max_iter
  for i=1:n
    % Evaluation step omitted
    [pbest,pbest_fit,gbest,gbest_fit]=checkFit(pbest,gbest,pbest_fit,
        gbest_fit,fit,i,swarm);
  end
  % Movement step
  w = (a-(a-b)*(k/max_iter));
  for i=1:n
    sum1 = c1*rand(1,d).*(pbest(i,:)-swarm(i,:));
    sum2 = c2*rand().*(gbest(1,:)-swarm(i,:));
    speed(i,:)=w*speed(i,:)+sum1+sum2;
    for j=1:d
      if speed(j)>Vmax(j)
        speed(j)=Vmax(j);
      end
    end
    swarm(i,:)=swarm(i,:)+speed(i,:);
    for j=1:d
      if swarm(i,j)<Lb(j)
        swarm(i,j)=Lb(j);
      elseif swarm(i,j)>Ub(j)
        swarm(i,j)=Ub(j);
      end
    end
  end
end
```

A.2.3.2 *Fit Checking for Particles*

This function check if a particle has bested his own personal best, or the global best of the entire population.

PSO fit checking

```
function [pbest,pbest_fit,gbest,gbest_fit] = checkFit(pbest,gbest,
    pbest_fit,gbest_fit,fit,i,swarm)
  gbest=gbest;
  if fit>pbest_fit(1,i)
    pbest_fit(1,i)=fit;
    pbest(i,:)=swarm(i,:);
    if fit>gbest_fit
      gbest=swarm(i,:);
      gbest_fit=fit;
    end
  end
end
```

B

THE BOUC-WEN MODEL OF HYSTERESIS

A hysteretic semi-physical model has been initially proposed by Bouc in 1971 [9] and subsequently generalized by Wen in 1976 [10]. Since then, it was known as the Bouc-Wen model and as been extensively used in the current literature to describe mathematically components and devices with hysteretic behaviours, particularly within the areas of civil and mechanical engineering.

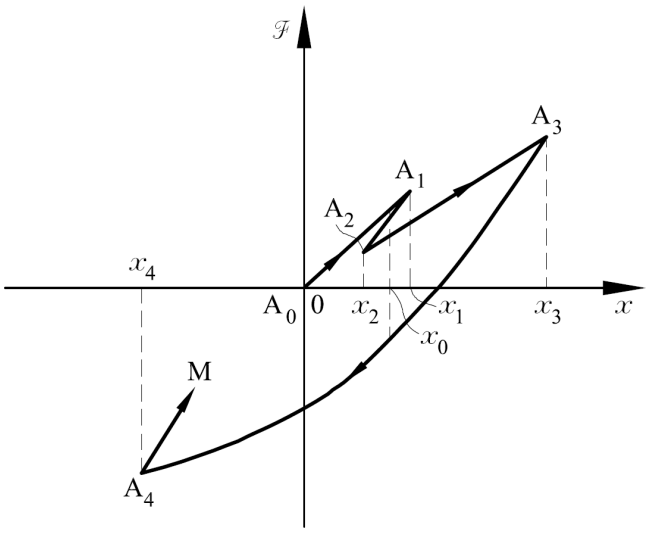

Figure 34: Graph force versus displacement for a hysteresis functional

The model consists of a first-order non-linear differential equation that relates the input displacement to the output restoring force in a hysteretic way. By choosing a set of parameters appropriately, it is possible to accommodate the response of the model to the real hysteresis loops.

In the first paper by Bouc, a functional that describes the hysteresis phenomenon was proposed. Given $\mathcal{F}$ a force and x a displacement, from Figure 34 it can be seen that four values of $\mathcal{F}$ correspond to the single point $x = x_0$. If we consider that x is a function of time, then the value of the force at instant time t depends not only from the value of the displacement x at time t, but also on past values of x.

The following simplifying assumption is made in the original paper.

Assumption B.1 *The graph of Figure 34 remains the same for all increasing function $x(\cdot)$ between 0 and x_1, for all decreasing function $x(\cdot)$ between the values x_1 and x_2, etc.*

In current literature, Assumption B.1 is referred to as the *rate-independent property* [27]. In his paper, Bouc proposes Equation 42 as a form for $\mathcal{F}$.

$$\frac{d\mathcal{F}}{dt} = g\left(x, \mathcal{F}, \operatorname{sgn}\left(\frac{dx}{dt}\right)\right)\frac{dx}{dt}. \tag{42}$$

Consider the equation

$$\frac{d^2x}{dt^2} + \mathcal{F}(t) = p(t) \tag{43}$$

for some given input $p(t)$ and initial conditions

$$\frac{dx}{dt}(t_0), \qquad x(t_0) \quad \text{and} \quad \mathcal{F}(t_0) \tag{44}$$

at the initial time t_0. Equations 42 and 43 describe completely a hysteretic oscillator. Since it is difficult to explicitly find a solution for Equation 42 due to the nonlinearity of the function g, Bouc proposes a variation of the Stieltjes integral to define $\mathcal{F}$:

$$\mathcal{F}(t) = \mu^2 x(t) + \int_\beta^t F\left(V_s^t\right) dx(s) \tag{45}$$

where $\beta \in [-\inf, +\inf)$ is the time instant after which the displacement and force are defined. V_s^t is the total variation of x in the time interval $[s, t]$. The function F is chosen so that it satisfies mathematical properties compatible with the hysteresis. In his work, Bouc chooses the function F in Equation 46:

$$F(u) = \sum_{i=1}^{N} A_i e^{-\alpha_i u}, \quad \text{with } \alpha_i > 0 \tag{46}$$

Using this result, Equations 43–46 can be rewritten in the following form.

$$\frac{d^2x}{dt^2} + \mu^2 x + \sum_{i=1}^{N} Z_i = p(t) \tag{47}$$

$$\frac{dZ_i}{dt} + \alpha_i \left|\frac{dx}{dt}\right| Z_i - A_i \frac{dx}{dt} = 0, \quad i = 1, \ldots, N \tag{48}$$

Equations 47 and 48 are known as the Bouc model. Equation 48 has been extended by Wen to describe the restoring forces with hysteresis in the following form:

$$\dot{z} = A\dot{x} - \alpha|\dot{x}|\, z^n - \beta\dot{x}\,|z^n| \qquad \text{for n odd} \tag{49}$$

$$\dot{z} = A\dot{x} - \alpha|\dot{x}|\, z^{n-1}\,|z| - \beta\dot{x}z^n \qquad \text{for n even} \tag{50}$$

Equations 49 and 50 are the earliest version of what is known as the Bouc-Wen model of hysteresis.

BIBLIOGRAPHY

[1] B. Tondu and P. Lopez, "Modeling and control of McKibben artificial muscle robot actuators," *IEEE control systems*, vol. 20, no. 2, pp. 15–38, 2000.

[2] P. J. Fleming and R. Purshouse, "Genetic algorithms in control systems engineering," 2001.

[3] X.-S. Yang, *Nature-inspired metaheuristic algorithms*. Luniver press, 2010.

[4] J. Kennedy, "Particle swarm optimization," in *Encyclopedia of machine learning*, Springer, 2011, pp. 760–766.

[5] C. P. Chou and B. Hannaford, "Measurement and modeling of McKibben pneumatic artificial muscles," *IEEE Transactions on robotics and automation*, vol. 12, no. 1, pp. 90–102, 1996.

[6] F. Al-Bender, V. Lampaert, and J. Swevers, "The generalized Maxwell-slip model: a novel model for friction simulation and compensation," *IEEE Transactions on automatic control*, vol. 50, no. 11, 2005.

[7] D. Lederer, H. Igarashi, A. Kost, and T. Honma, "On the parameter identification and application of the Jiles-Atherton hysteresis model for numerical modelling of measured characteristics," *IEEE Transactions on magnetics*, vol. 35, no. 3, pp. 1211–1214, 1999.

[8] P. Ge and M. Jouaneh, "Generalized Preisach model for hysteresis nonlinearity of piezoceramic actuators," *Precision engineering*, vol. 20, no. 2, pp. 99–111, 1997.

[9] R. Bouc, "Modèle mathématique d'hystérésis : application aux systèmes à un degré de liberté," *Acustica*, 1969.

[10] Y. K. Wen, "Method for random vibration of hysteretic systems," *Journal of the engineering mechanics division*, vol. 102, no. 2, pp. 249–263, 1976.

[11] C. P. Chou and B. Hannaford, "Static and Dynamic Characteristics of McKibben Pneumatic Artificial Muscles," in *Proceedings of the 1994 IEEE International Conference on Robotics and Automation, San Diego, CA*, 1994.

[12] H. F. Schulte, "The Characteristics of the McKibben Artificial Muscle," in *The Application of External Power in Prosthetic and Orthotics*, National Academy of Sciences - National Research Council, Ed., 1961.

[13] *System Identification Toolbox, Create linear and nonlinear dynamic system models from measured input-output data*, 2018. [Online]. Available: https://uk.mathworks.com/products/sysid.html.

[14] F. Preisach, "Über die magnetische Nachwirkung," *Zeitschrift für Physik*, vol. 94, no. 5, pp. 277–302, 1935, ISSN: 0044-3328. DOI: 10.1007/BF01349418. [Online]. Available: https://doi.org/10.1007/BF01349418.

[15] F. Al-Bender, V. Lampaert, and J. Swevers, "The generalized Maxwell-slip model: a novel model for friction simulation and compensation," *IEEE Transactions on automatic control*, vol. 50, no. 11, pp. 1883–1887, 2005.

[16] M. A. Krasnosel'skii and A. V. Pokrovskii, *Systems with hysteresis*. Springer Science & Business Media, 2012.

[17] M. Ismail, F. Ikhouane, and J. Rodellar, "The hysteresis Bouc-Wen model, a survey," *Archives of Computational Methods in Engineering*, vol. 16, no. 2, pp. 161–188, 2009.

[18] J. Song and A. Der Kiureghian, "Generalized Bouc–Wen model for highly asymmetric hysteresis," *Journal of engineering mechanics*, vol. 132, no. 6, pp. 610–618, 2006.

[19] W. Kobayashi, K. Ito, and S.-i. Yamamoto, "Displacement Control of Water Hydraulic McKibben Muscles with Load Compensation," *JFPS International Journal of Fluid Power System*, vol. 8, no. 2, pp. 107–112, 2014.

[20] W. Kobayashi and K. Ito, "Displacement estimation of tap-water driven mckibben muscles," in *2015 International Conference on Fluid Power and Mechatronics (FPM)*, IEEE, 2015, pp. 672–676.

[21] W. Kobayashi, K. Ito, and P. Zobel, "Development of gait-training orthosis with water hydraulic McKibben muscle," in *Proc. of the 12th International Conference on Fluid Control, Measurements, and Visualization*, vol. 3, 2013.

[22] M. A. Zaman and U. Sikder, "Bouc–Wen hysteresis model identification using modified firefly algorithm," *Journal of Magnetism and Magnetic Materials*, vol. 395, pp. 229–233, 2015.

[23] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on*, IEEE, 1995, pp. 39–43.

[24] F. Ikhouane, V. Mañosa, and J. Rodellar, "Dynamic properties of the hysteretic Bouc-Wen model," *Systems & control letters*, vol. 56, no. 3, pp. 197–205, 2007.

[25] dSPACE, *DS1104 Controller Board*, 2019. [Online]. Available: https://www.dspace.com/en/inc/home/products/hw/singbord/ds1104.cfm.

[26] J. Mattingley and S. Boyd, "CVXGEN: A code generator for embedded convex optimization," *Optimization and Engineering*, vol. 13, no. 1, pp. 1–27, 2012.

[27] A. Visintin, *Differential models of hysteresis*. Springer Science & Business Media, 2013, vol. 111.