UNIVERSIDAD POLITÉCNICA DE PUEBLA

PROGRAMA ACADÉMICO DE POSGRADO

Sintonización de un controlador PID de orden fraccional a partir de Redes Neuronales de Base Radial y su Implementación con Arreglos Analógicos Programables FPAA

TESIS QUE PARA OBTENER EL GRADO DE

MAESTRÍA EN INGENIERÍA EN AUTOMATIZACIÓN DE PROCESOS INDUSTRIALES

PRESENTA:

LUIS VLADIMIR GARCÍA JIMÉNEZ

Director: Dr. Carlos Muñiz Montero
Co-Director: Dr. Luis Abraham Sánchez Gaspariano

Juan C. Bonilla, Puebla, Mexico, Junio 2017.

El presente trabajo fue realizado en el Laboratorio de Investigación y Posgrado de la Universidad Politécnica de Puebla, ubicada en Tercer carril del Ejido "Serrano" S/N, San Mateo Cuanalá, Municipio Juan C. Bonilla, Puebla CP 72640.

Apoyo del CONACYT, Beca No. 636153, Programa de Maestría perteneciente al Programa Nacional de Posgrados de Calidad (PNPC-CONACYT). Apoyo del CONACYT a través del Proyecto Ciencia Básica 181201.

CONACYT

Universidad Politécnica de Puebla

Maestría en Ingeniería en Automatización de Procesos Industriales

Sintonización de un controlador PID de orden fraccional a partir de Redes Neuronales de Base Radial y su Implementación con Arreglos Analógicos Programables FPAA

Tesis realizada por:

LUIS VLADIMIR GARCÍA JIMÉNEZ

Aprobada por ... Septiembre 14, 2017.

Profesor **(Firma)**

Dr. Carlos Muñiz Montero ....

Dr. Luis Abraham Sanchez Gaspariano ....

Dr. Javier Lemus Lopez ....

Dr. Ernesto Castellanos Velasco ....

Juan C. Bonilla, Puebla, Mexico, Septiembre 2017.

UNIVERSIDAD POLITÉCNICA DE PUEBLA
MAESTRÍA EN INGENIERÍA EN AUTOMATIZACIÓN DE PROCESOS INDUSTRIALES

(Firma)

....
Luis Vladimir García
Jiménez

Abstract

A partir de las dificultades que implica el uso de herramientas del cálculo no entero o fraccional aplicados al modelado y control de sistemas dinámicos, se resuelven dos limitantes en el diseno de sistemas de control de tipo fraccional, la sintonización e implementación de controladores $PI^{\lambda}DI^{\mu}$.

En este trabajo se propone una solución a través de circuitos analógicos programables y reconfigurables al reto de implementar físicamente controladores PID de orden fraccional, se muestra también una herramienta accesible a estudiantes e ingenieros (MATLAB) para realizar la sintonización de este tipo de controladores.

Currently the PID controller is the most widely used in industry due to its simple structure and relatively easy understanding. This work proposes the introduction of a fractional order PID controller to obtain a more flexible and robust control than the conventional PID, with the inclusion of two additional degrees of freedom. The $PI^{\lambda}DI^{\mu}$ controller has an integrator of order λ and a differentiator of order μ, with λ, $\mu \in (0, 2)$ allowing to add restrictions and specifications for the tuning process, improving the performance of the system response. In addition, it is proposed the use of reconfigurable devices, such as FPGAs or FPAAs, to make changes *in situ*. It is also proposed the use of neural networks to estimate the values P, I, D, λ and μ in order to make the system insensible to variations in the plant parameters.

Keywords

Controlador PID fraccional, implementación de operador de Laplace no entero, PID fraccional analógico, aproximación de derivada e integral no entera.

Agradecimientos

This work was supported by the National Council for Science and Technology (CONACyT), Mexico, under Grant 181201. Additionally, this work was also partially sponsored by projects: CONACYT-Project No. 258880, VIEP-BUAP, Plan de Trabajo CA (BUAP-CA-276), and SEP-PRODEP.

Simplicity is the ultimate sophistication. Da Vinci.

Éste es un trabajo personal más bien colectivo... con aprecio para: Asesores, maestros, etc.

Luis Vladimir García Jiménez

Contenido

Lista de Figuras

Lista de Tablas

I

Research approach

Chapter 1: Research approach

1.1 Introduction

The time and frequency response of electronic, mechanical or electromechanical systems widely used in various industries is often expected to improve. It can even occur that during any time of the operation those systems face unforeseen circumstances within its range of action and being necessary to compensate its operation to obtain greater speed, greater stability, lower energy costs, etc. To have superior performance subsystems are incorporated in place to compensate the frequency or temporal response avoiding abrupt changes to unforeseen entries, maintaining oscillations as small as possible or responding quickly but within an acceptable range. These systems are known because of their function as compensators or controllers and are composed mainly of mechanical, electronic or embedded systems.

Among many control strategies, such as lead-lag networks [2], *root locus* , sliding modes [3] and others, the Proportional Integral Derivative control algorithm (PID) is the most used in the industry due to its simple structure and its relatively easy understanding . Nevertheless, many real physical systems are modeled more accurately from differential equations of fractional or non integer order instead integer ones, therefore fractional order control techniques have been explored to produce a more flexible and robust control of these systems than the conventional PID [4]. Thus in 1997 Podlubny [5] proposed a fractional order generalization of the PID controller which included an integrator of order λ and a differentiator of order μ where $\lambda, \mu \in (0, 2)$. The incorporation of this two additional degrees of freedom in the control design process allowed to add more specifications and restrictions to the system response improving the performance through the process of tuning proportional, integral and derivative gains as well as the non integer order parameters λ and μ.

Unfortunately, the main disadvantage of the fractional-order PID controller (FOPID) is that its analog implementation is still an open problem because the reported implementations are rather bulky and hard to accomplish with commercial component values of resistors, capacitors and inductors [6–8]. Besides, the tuning of its five parameters is an optimization problem [9].

1.2 Description of the problem

In the literature, several fractional PID controller parameters tuning techniques have been reported [9–17], which allow to pose the problem of control as a set of simultaneous nonlinear equations [9]. Unfortunately, those procedures suppose a complete knowledge of the parameters of the plant. Even more, if the plant features changing conditions, then it is necessary to re-characterize these parameters, rethink the system of equations and modify some hardware related to the controller. A solution to the problem of changing plant conditions was proposed in [1], where the fractional PID controller was tuned from the training of Radial-Based Neural Networks (RBNN), one for each of its five parameters:

proportional gain K_P, integral gain K_I, derivative gain K_D, λ the fractional order of the signal error integral y μ fractional order of the signal error derivative. In [1] restrictions were set up on the system response composed by the plant and the fractional PID controller to carry out the training of the neural networks. These restrictions were: (1) margin gain, (2) margin phase, (3) high frequency noise rejection, (4) output disturbance rejection (sensitivity) and (5) robustness to variations in the plant gain.

The chosen plant was modeled with a pole and a term for the transport delay. Finally, simulation results were showed. Unfortunately, the solution reported in [1] was only validated by means of simulation results. The same happens with other reported works (for instance, [12, 17, 18]). In consequence, the problems related to a possible implementation of the designed FOPID are not addressed. This difficulty stems from the fact that there are no circuit elements with fractional order response (for example, the so-called "fractal capacitors"). Some authors propose to make approximations of fractional integrators with rational functions of the Laplace variable 's', however, this practice results in bulky circuits with capacitor and resistor arrays with values difficult to acquire commercially. The problem is aggravated when considering a plant with changing characteristic since this approach of fractional integrators would have to be redesigned which totally changes those arrays.

1.3 Proposed solution and possibility of contribution

In Chapter 2 of this thesis are presented the states-of-the-art respect to tuning methods of the parameters of the FOPID and respect to analog implementation of fractional order systems. From those analysis, and starting from the reported work in [1], it is proposed a solution for a practical implementation of fractional order PID controller by means of opamp-based building blocks and by mean of reconfigurable hardware. According to Figure 1.1, a model of a plant and the FOPID control will be implemented in hardware from development boards with field-programmable analog arrays (FPAA) model AN231E04 from the manufacturer Anadigm. For this, a methodology is proposed for the implementation of fractional order integrators and differentiators, just modifying the gains of integral and differential amplifiers. This methodology will allow to establish the orders of the derivative and integral fractional modifying only the gains of the differential amplifiers, thus avoiding the need for arrays of resistor and capacitor whose commercial values are not common or difficult to acquire.

Moreover, being electrically reconfigurable, the use of FPAAs will not stop the system function or make changes to the hardware when there is a need to compensate plant variations. These variations may also be modeled and electrically established with the FPAA for system testing purposes. Neural networks will be trained offline with the criteria set out in [1]. A solution to perform both the optimization of values (tuning) as to implement the RBF neural network with minimal errors is to use robust tools such as the MATLAB computer program.

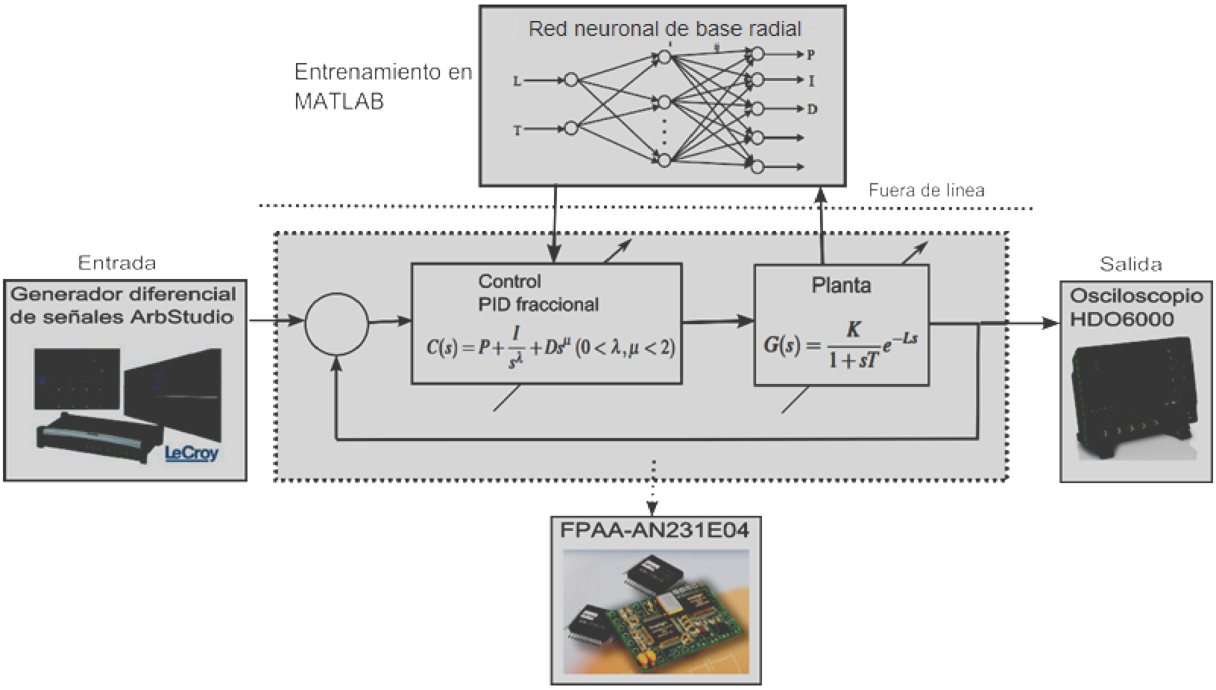

Red neuronal de base radial
Entrenamiento en MATLAB
L
T
P
I
D
Fuera de linea
Entrada
Generador diferencial de señales ArbStudio
LeCroy
Control PID fraccional
$C(s) = P + \frac{I}{s^{\lambda}} + Ds^{\mu}\ (0 < \lambda, \mu < 2)$
Planta
$G(s) = \frac{K}{1+sT}e^{-Ls}$
Salida
Osciloscopio HDO6000
FPAA-AN231E04

Figure 1.1: *Fractional PID plant-controller system with negative feedback.*

1.4 Methodology

In the laboratory of the Academic Body of Electronics CA-3 from Universidad Politécnica de Puebla, we had the following infrastructure, sufficient for the realization of this project: eight development boards AN231E04, a high-resolution oscilloscope (12 bits) HDO6000 at 300MHz, an arbitrary signal differential generator ArbStudio, amplifiers, attenuators and differential tips, a data acquisition card ELVIS-II among other digital and analog instrumentation.

For the training of neural networks and the simulation of the complete system we used Simulink of MATLAB and HSPICE for simulation at the circuit level. Anadigm Designer 2 software was also used for the programming and simulation of the FPAA.

1.5 General objective

To develop a methodology to implement with both, opamp-based circuits and FPAAs, first-order plus dead-time plants controlled by means of fractional-order PID controllers, which are tuned by means of RBNN in order to obtain reconfigurable systems, robust to tolerances in the plant parameters and avoiding the use of non-commercially available passive elements.

Implement a fractional order PID control system, in field-programmable analog arrangements FPAA-AN231E04, tuned from radial-based neural networks trained offline with MATLAB.

1.6 Particular objectives

1. To perform a comparative study of the fractional PID controllers reported in the literature for the determination of the plant and the tuning techniques of the controller to be implemented.

2. To propound, based on the characteristics of the plant and the specifications and design constraints of the control system, the simultaneous nonlinear equations to solve with MATLAB to establish the values of the parameters P, I, D, λ, μ of the controller.

3. To perform the training of radial-based neural networks in MATLAB to obtain the data set P, I, D, λ, μ under changing conditions.

4. To model the plant and the fractional PID controller in the form of electronic circuits for conducting physical tests and characterization from field programmable analog arrays FPAA231E04 and opamp-based circuits.

1.7 Justification

The implementation of a fractional system, either the plant or its control, imposes the need to develop a method for adjusting physically the values λ y μ in the most agile way. From the two general approaches, one digital and the other analog, requirements for implementation are revealed, e.g., the digital makes use of a microprocessor and an appropriate control algorithm while analogue implementation involves the application of analog circuits and fractal elements which are approximated actually by bulky and complicated arrangements of resistors and capacitors with values difficult to acquire commercially.

Not having affordable fractal elements to anyone interested in deploying non-integer order systems makes it difficult to penetrate fractional PID control in industry, that is, the characteristics associated with the configuration of fractal elements prevent control engineers from taking advantage of the fractional PID control over conventional PID control. Therefore, the development of a simple methodology to implement in reconfigurable hardware the fractional PID control will provide the conditions to take advantage of a more flexible and robust control.

Thus, the implementation proposal translates the use of arrays made up of ordinary passive elements of fractional behavior to the adjustment in the gains of differential amplifiers and integer order integrators to set the non -integer specified order, so that, will allow the more agile and productive experimental development of fractional order control systems that provide superior control over the responses in time and frequency than integer order systems.

Furthermore, the use of neural networks will allow an easy adjustment of the control system in case that significant changes are detected in the plant, without making changes to the implementation in hardware, thanks to the on-site programming of the FPAAs.

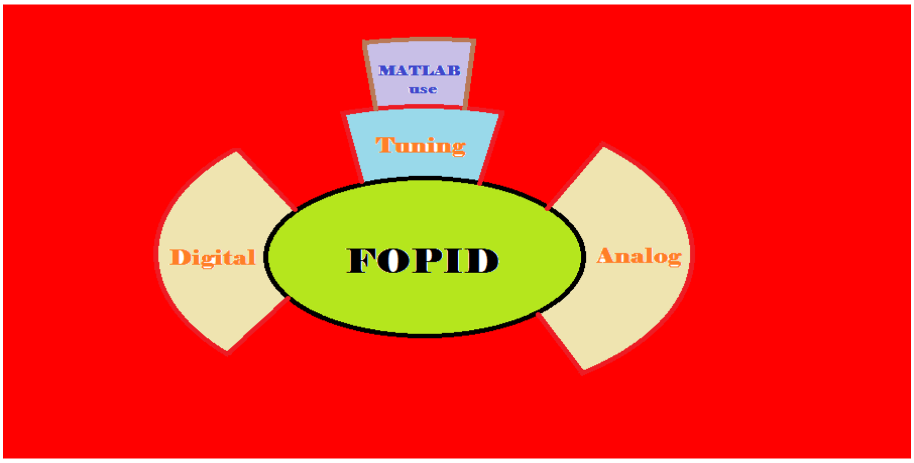


Figure 1.2: *Fractional-order PID controller divided into its main parts.*

1.8 Structure of the thesis

In the next four chapters there is an approach for every aspect of the proposed solution. In Chapter 2 it is founded the fundamental concepts of fractional calculus and PID controller as the state of the art in the implementation of fractional systems, the fractional PID controller and tuning techniques. In Chapter 3 the proposed solution is presented through an analog method of approximation to develop fractional operators. On Chapter 4 the proposed tuning method is described. In Chapter 5 there is an example of applications in the field of filters using the proposed method. Figure **??** shows a division of how the fractional order PID controller is seen in this work, according to the parts are to be covered, analog implementation, tuning and the use of computational tools (MATLAB).

II

Theoretical background and state of the art

2.1 Derivative and integral of fractional order

The fractional calculus is a generalization of the differentiation and integration of arbitrary order, not precisely integer. A more correct term to call it should be non-integer order calculus since the order may be even irrational or imaginary [19, 20], but keeping it simple the most common meaning has already been given: fractional calculus.

There are several definitions of the derivative and integral of non-integer order that show the existence of both fundamental operators of arbitrary order and are basically obtained using the conventional definitions of integrals and derivatives repeatedly in an iterative process, in the analogous sense to what a notation of exponents represents for multiplication of a numerical value by itself as $x^2{=}x * x$. Geometric interpretation or physical meaning of each fractional operator is not the subject of this thesis nor will be detailed in this document. Among all definitions for calculating fractional derivatives and integrals, Riemann-Lieuville definition establishes [20]

$$D_t^{\alpha} f(t) = \frac{1}{\Gamma(m-\alpha)} \left(\frac{d}{dt}\right)^m \int_0^t \frac{f(\tau)d\tau}{(t-\tau)^{\alpha-m+1}} \tag{2.1}$$

where $\alpha \in \mathbb{R}$, $m-1{<}\alpha{<}m$, $m \in \mathbb{N}$ and $\Gamma(\cdot)$ is Gamma function. For $\alpha{>}0$, $\alpha{<}0$ and $\alpha{=}0$ one gets the fractional derivative, integral and identity function.

2.1.1 Properties

The properties of the fractional operator are as follows [**?**]:

1. If $f(t)$ is an analytical function, then the fractional order operator ${}_a\mathscr{D}_t^{\alpha}$ is also analytical with respect to t.

2. If $\alpha{=}n$ y $n \in \mathbb{Z}^+$, then the operator ${}_0\mathscr{D}_t^{\alpha}$ can be understood as the usual operator $\dfrac{d^n}{dt^n}$.

3. The operator or order $\alpha{=}0$ is the identity operator ${}_0\mathscr{D}_t^0 f(t){=}f(t)$.

4. Fractional order differentiation is linear; if a, b are constant, then

$$ {}_0\mathscr{D}_t^{\alpha}[af(t)+bg(t)] = a\,{}_0\mathscr{D}_t^{\alpha} f(t) + b\,{}_0\mathscr{D}_t^{\alpha} g(t)$$

5. For fractional order operators with $R(\alpha){>}0$, $R(\beta){>}0$, and under reasonable restrictions on the function $f(t)$ the additive law of exponents is preserved

$$ {}_0\mathscr{D}_t^{\alpha}\left[{}_0\mathscr{D}_t^{\beta} f(t)\right] = {}_0\mathscr{D}_t^{\beta}\left[{}_0\mathscr{D}_t^{\alpha} f(t)\right] = {}_0\mathscr{D}_t^{\alpha+\beta} f(t)$$

6. The fractional order of the derivative commutes with the derivative of integer order

$$\frac{d^n}{dt^n}\left[{}_a\mathscr{D}_t^{\alpha} f(t)\right] = {}_a\mathscr{D}_t^{\alpha}\left[\frac{d^n}{dt^n} f(t)\right] = {}_a\mathscr{D}_t^{\alpha+n} f(t),$$

under the condition t=a, $f^k(a)$=0 for k integer greater than or equal to zero and less than n.

2.1.2 Laplace fractional operator and transfer function

The Laplace transform is an essential tool in engineering and control systems. A function $F(s)$ of the complex variable s is called the Laplace transform of the original function $f(t)$ and is defined as

$$F(s) = \mathscr{L}[f(t)] = \int_0^{\infty} \frac{f(\tau)}{(t-\tau)^{a-m+1}} d\tau \tag{2.2}$$

The original function $f(t)$ can be recovered from the Laplace transform $F(s)$ by the application of the inverse Laplace transform defined as

$$f(t) = \mathscr{L}^{-1}[F(s)] = \frac{1}{2\pi j} \int_{c-j\infty}^{c+j\infty} e^{st} F(s) ds \tag{2.3}$$

with c greater than the real part of all poles of the function $F(s)$.

Laplace Transform with initial conditions equal to zero of (2.1) is given by [20]

$$L\{D_t^a f(t)\} = s^a F(s) \tag{2.4}$$

where s^a is the Laplace operator of fractional order expressed as [20]

$$s^a = (j\omega)^a = \omega^a \left[\cos\left(\frac{a\pi}{2}\right) + j\sin\left(\frac{a\pi}{2}\right)\right] \tag{2.5}$$

2.2 Analogue implementation of fractional order Laplace operators

The challenge in implementing fractional-order systems is related to the non-existence of circuit elements that reproduce the operator s^a, therefore it is necessary to have good approximations of 2.5 with a minimum of parameters. Preferably approximations or equivalences in rational form, with poles and zeros. There are Power Series Expansion (PSE), Continuous Fraction Expansion (CFE), discretization by the recursive method of Muir, among others [6, 7, 18]. Regarding the analog implementation, there are basic devices that fulfill the purpose of approaching a fractional behavior. The most popular is the network with ladder circuits [6, 18, 21]; the other consists of electrical elements with a tree structure and the transmission line circuit [18]. These devices are constituted by branches of the three ordinary passive electrical elements: resistor, inductor and capacitor and sometimes it require negative impedance converters. Two other concepts for implementation are through Warburg's impedance built with impedances in ladder structure [18] and integrating operational amplifiers [6, 18], and the fractor [18], which is a new electrical element with fractional frequency-dependent impedance made of special materials, composites. The most widespread method among different authors is to use

$$Z = z_1 + \cfrac{1}{y_2 + \cfrac{1}{z_3 + \cfrac{1}{y_4 + \ldots}}}$$

$$\begin{aligned}
Z &= \frac{(s+2)(s+4)}{(s+1)(s+3)} = \frac{s^2+6s+8}{s^2+4s+3} = 1 + \frac{2s+5}{s^2+4s+3} \\
&= 1 + \cfrac{1}{\cfrac{s^2+4s+3}{2s+5}} = 1 + \cfrac{1}{s/2 + \cfrac{3s/2+3}{2s+5}} \\
&= 1 + \cfrac{1}{s/2 + \cfrac{1}{\cfrac{2s+5}{3s/2+3}}} = 1 + \cfrac{1}{s/2 + \cfrac{1}{4/3 + \cfrac{1}{3s/2+3}}} \\
&= \boxed{1} + \cfrac{1}{\boxed{s/2} + \cfrac{1}{\boxed{4/3} + \cfrac{1}{\boxed{3s/2} + \boxed{\frac{1}{1/3}}}}}
\end{aligned}$$

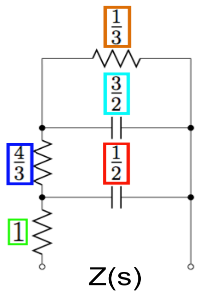


Figure 2.1: *Method of Cauer for circuit synthesis.*

the Oustaloup filter to perform the approximation of the operator s^a in the frequency domain [7, 18].

Analog approximation of fractances are characterized by a magnitude response with roll-off ±20a decibels by decade, and a constant-phase response at all frequencies of ±90a degrees [7]. For example, a fractal capacitor has impedance Z=1/($s^{\partial}C$). The fractances can be approached in a desired bandwidth with rational functions from the methods of Newton, Muir, Oustaloup, Matsuda, power series expansion (PSE), continuous fractions expansion (CFE), among others [7, 18]. Once a rational function is obtained it can be synthesized with ladder networks, tree structure, or transmission lines [7, 18] [**?**]. The circuit components can be resistors, inductors, capacitors and sometimes negative impedance converters [6, 7, 18, 22]. One example for synthesis by Cauer method is given in Fig. 2.1.

Ladder networks by synthesis of electrical circuits were developed almost a century ago in the 1920's by Cauer and Foster. Its canonical forms use LC networks which find the coefficients by a continuous fraction expansion. When using just two kinds of elements, the remaining impedance must be positive real but if RLC elements are used negative impedances could appear [18].

The works of Bode y Antoniou left some advances in the use of operational amplifiers after 1960, that mixed with non integral impedances made possible active fractional devices. Also, Dutta Roy developed in those years constant-argument immitance (impedance or admittance) [18]

In 2002 Podlubny presented an approach using analogue circuits for the implementation of fractional-orders controllers, compiling Carlson, Matsuda and Oustaloup methods, among other CFE like methods to develop building blocks of fractional operators [6].

In 2006 there were published works of Biswas and Tenreiro showing some advances in the fabrication of electrochemical devices with fractal response [**?**, 23]. This devices are complex and even not very developed to make it useful in the design of integrated circuits.

During the beginning of this decade were published some compilations of the new tendencies in the field of both, digital and analog implementations. Chen and Krishna are good examples of this initiative [7, 18].

Method	Includes	Advantages	Disadvantages
Rational approximation [6–8, 18, 21]	Cauer Foster Carlson Matsuda Oustaloup Tree type Net grid	Stability Few iterations Convergence in large domains Work for arbitrary orders	Computed values not usually available Sometimes needs negative impedances Inductors are bulky
Transmission line [18]	Warburg	Use identical R-series, C-shunt	Fixed S^{a} or $S^{-}a$
CPE [18, 23, 25]	Capacitive probe (difussion)	Chemically designed	Expensive Bulky
Active [6, 18, 21, 22, 24, 26]	NIC (Z + OpAmps) FO capacitor/inductor	Cover negative impedance	Power supply needed

Table 2.1: *Comparison of the implementations of fractional s^{a} operators.*

Recent studies are more focused on non linear behavior in networks, biological systems and its interconnection. One interesting study over recent technology is found in Pu's work [24].

2.3 PID

When the dynamics of a system or its components is known then is possible to model its behavior and know how it will act under certain circumstances. Under this assumption, corrective actions can be applied. However it is frequent to find that it is unknown some part or practically all the nature of the systems. In consequence, despite the abundance of today's sophisticated tools, a type of control in which engineers in many industries rely on is the proportional, integral and derivative controller, the PID controller [27]. It is estimated that more than 90% processes involving feedback have one of these or some of its variants [18, 27]. In general from the PID control the following functions can be emphasized: it provides feedback, it has the ability to eliminate steady state errors through integral action and can anticipate future states through derivative action [27].

In 1922, technical considerations related to PID control began to be taken more rigorously. In the decade of the 40's some studies lead to established rules for the PID tuning by means of the method of Ziegler-Nichols and since then there are numerous empirical or analytical approaches to their tuning.

The PID controller scheme is shown in Figure 2.2 (a). It is formed by blocks with an input leading to the controller with a power stage, the plant and an output. The whole block is feedback by a sensor that extracts the signal of exit to lead it to the entrance and to close the loop. The framed block at the beginning generates its control action

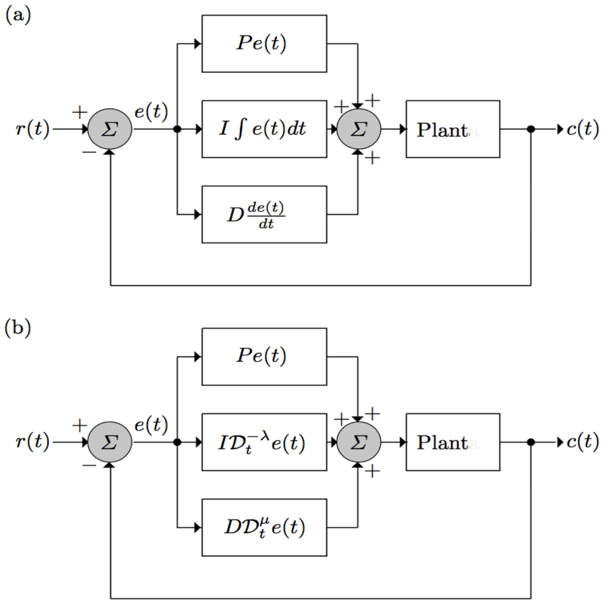


Figure 2.2: *(a) Proportional-Integral-Derivative controller (PID). (b) PID controller generalized to fractional order (PI$^{\lambda}$D$^{\mu}$ controller).*

according to the error derived from extracting the difference between the input and the output driven by the sensor. Therefore, the analog PID equation is

$$u(t) = K_P e(t) + K_I \int e(t)dt + K_D \frac{e(t)}{dt} \tag{2.6}$$

where K specifies proportional gain, T_I characterizes the integral action, also known as the integral-time constant and T_D represents the derivative action and is known as the derivative time constant. The integral term of the PID controller is $T_I = 1/K_I$ and is defined as the period for which the integrating effect on the error is equivalent to the proportional action. The derivative term of the PID controller is $T_D = K_D$ and is defined as the period for which the differentiating effect on the error is equivalent to the proportional action.

Due to the proportional action, a comparison of the output is made with the value fixed at the input (setpoint), the integral action eliminates the steady state error when the reference is constant and the derivative action reacts to large change rates before the error takes up high values. Table 2.2 shows the effects exerted by each action on the general behavior of the control system. In general terms the PID control is robust to variations when all three parameters are correctly tuned [27].

It is important to note when the PID controller can be used in a process and when it is not convenient. If the performance requirements are not very strict this controller can be included and also when the system is of second order or in rigid systems; The PI version can be applied when the system dynamics is of first order or higher order but sufficiently damped and at low frequencies or with reduced bandwidth, some engineers or operators prefer to cancel the effect of the derivative action due to difficulties for precise tuning. On the contrary, it is not advisable to apply it when dealing with higher order systems, with long dead times or systems with oscillatory modes [27].

	Rise time	Maximum override	Settle time	Steady state error
P	Decrease	Grow	Small changes	Decrease
I	Decrease	Grow	Grow	Elimination
D	Small changes	Decrease	Decrease	Small changes

Table 2.2: *Effects of each control action on the system.*

2.4 $PI^{\lambda}D^{\mu}$ controller

The first reported application of fractional calculus was in physics when Abel used it to solve the problem of tautochronous movement, more recently applications in mechanics and electrical have been published [14, 18, 26, 28]. Due to the absence of appropriate mathematical methods, non-integer order dynamical systems were studied marginally in theory and applications in control systems until in the last decade of the twentieth century Oustaloup proposed the application of fractional order controllers in dynamic systems and in this way the controller CRONE was developed demonstrating a superiority against the conventional PID [18,29,30]. Later Podlubny proposed the fractional order PID controller model, also showing the advantages over the ordinary PID [5,6]. On the other hand, there are two more controllers that represent also non-integer controllers, these are the TID controller (Tilt-Integral-Derivative) and the fractional lead-lag compensator [2, 18].

It is possible to generalize the *PID* controller from Figure 2.2 (b), to the fractional order $PI^{\lambda}D^{\mu}$ controller shown in Fig. 2.2(b) [5, 6]. This controller realizes integral and derivative actions of orders $\lambda \in \mathbb{R}^{+}$ and $\mu \in \mathbb{R}^{+}$, to the signal error $e(t)$, in addition to the proportional action, which is defined by the integral-differential equation

$$u(t) = Pe(t) + ID_t^{\lambda}e(t) + DD_t^{\mu}e(t), \quad (\lambda, \mu > 0) \tag{2.7}$$

which corresponds to the transfer function

$$C(s) = \frac{U(s)}{E(s)} = P + \frac{I}{s^{\lambda}} + Ds^{\mu} \tag{2.8}$$

where P denotes the proportional gain, I the integral gain, and D the derivative gain. As mentioned above, the $PI^{\lambda}D^{\mu}$ controller has better performance than the *PID* one because the former has five parameters (P, I, D, λ, μ) that allows us to establish five design restrictions that will been explained in Section 2.5.1. These parameters can be tuned in the frequency domain, as shown in [9, 31]. In this work we discuss current methods to approach fractances in the realization of analog $PI^{\lambda}D^{\mu}$ controllers.

For the case λ=μ=1 we have the traditional PID, if λ=1 y μ=0 this is the PI variant of the controller and if λ=0 with μ=1 a typical PD controller is obtained, that is, the latter two are particular cases of the general controller. This allows to deduce that the variation of the parameters λ and μ will give greater flexibility to the adjustment of the dynamic properties of the fractional control system. Figure 2.3 illustrates some variants corresponding to the

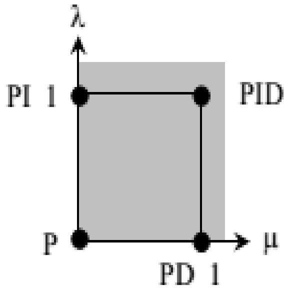


Figure 2.3: *Plain $\lambda - \mu$ where the exponent of the fractional operator is located.*

classical PID controller as a subset of the coordinate system, where there is an infinite number of pairs leading to different specifications in the controller response.

2.5 Tuning or adjustment methods

Once the transfer function or model in the frequency domain of the fractional controller is known, it is necessary to specify design methods to meet the control objective. The classical methods for tuning the integer PID have not been discarded at all, for example, the influence that the Ziegler-Nichols method has had on the design of the traditional PID, although to a lesser extent, is still present for the case $PI^{\lambda}D^{\mu}$ despite the existence of more refined methods [27].

The Ziegler-Nichols method is empirical, based on the simulation of a large number of cases and abridged to two rules [27, 32], one applicable to the plant response over time and the other observing the frequency response. The first rule establishes that for a unit step input two parameters of the plant in open loop are measured: the delay time L and the time constant T, knowing both of them is determined by means of a table which values have to take the parameters of the controller, K_P, T_I and T_D.The second rule states that nullifying terms T_I and T_D of the controller and establishing a feedback loop, K_P will be increased from 0 to a critical value K_c where the system response begins to oscillate, measuring the period P_c of sustained oscillations, again these two parameters allow us to determine which values have to take each parameter of the controller, K_P, T_I and T_D through its corresponding table [27, 33].

In the case of $PI^{\lambda}D^{\mu}$ similar processes are used, the simplest is to arbitrarily propose the values of the two additional parameters to the integer-order PID, e.g. λ=μ=0.5 [6, 18], this method clearly misses the adjustment of the two degrees of freedom that a design practice can lead to a better performance of the new controller. Another more elaborated method specifies a requirement in the frequency domain, the phase margin φ_{pm}, then it gets the terms K_P, K_I through the Ziegler-Nichols method plus solving a pair of simultaneous equations and using the method of Astrom-Hagglung the value K_D is proposed, finally, two simultaneous equations involving K_P, K_I y K_D bring the remaining two parameters [13]. Another one [34], tunes the traditional PID out from the parameters used by excellence in determining the stability of a system in the frequency domain: the

Method	Includes	Advantages	Disadvantages
Rule-based [11, 36]	Z-N kind (2-DOF)	Previous known method	Non optimal tuning
Analytical [9, 10, 13–15, 36, 43, 44]	PM - GM Flat phase IMC Dominant poles F-MIGO PM - GM - robustness	Medium-high accuracy	Previous model knowledge Complex
Numerical [3, 40, 40–42, 45–48]	GA PSO Fuzzy ANN	High accuracy	Computationally expensive Very complex

Table 2.3: *Comparison of the fractional PID tuning techniques.*

phase margin ϕ_{pm} and the gain margin K_g, used to find numerically λ y μ. Within the analytical tuning methods used for the integer PID, the method λ, useful in systems with lengthy dead time L, the method of Haalman or the IMC (internal mode controller) which cancels transfer function poles and is an extension of Smith's predictor [27]. For the case of pole location, the Cohen-Coon method is used to place the dominant poles of the system at will [27, 35].

Finally, optimization techniques are valid also for tuning purpose and its use has been extended [1, 36], is based on the specification of parameters expressed as function inequalities where the most important or significant specification is chosen as the function to be optimized. The algorithms that determine when the expected value has been reached are evaluated by means of a gradient: ISE, IAE, ITAE, ITSE [37].

There are three possible ways to optimize parameter values K_P, K_I, λ, K_D y μ of the controller, one is optimizing gains K_P, K_I y K_D, another is to optimize only the orders λ y μ or all, simultaneously [38].

In the past decade few papers tackle the problem of fractional PID tuning but there was a clear tendency to solve it using optimization techniques, Monje *et al* and Biswas were representative examples of the use of mixed approaches, analytical plus numeric [9].

Since 2010, there are more works on artificial neural networks (ANN), particle swarm optimization (PSO), genetic algorithms (GA) and fuzzy logic whose focus is on minimization of a target function or, in general, optimization process [1, 3, 39–42]. Many works parts from the frequency domain analysis, and a few from the time domain analysis.

2.5.1 Monje *et al* adjustment method

In a paper published in 2004, 5 performance specifications are proposed in the frequency domain to obtain the 5 unknown parameters of the fractional controller or, in other words, tune the controller. The conditions are as follows [9]:

1. Phase margin

$$|C(j\omega_{cg})G(j\omega_{cg})|_{\omega_{cg}=\frac{1}{2}\frac{rad}{s}} = 0dB \tag{2.9}$$

2. Gain margin

$$arg(C(j\omega_{cg})G(j\omega_{cg})) = -\pi + \phi_m \tag{2.10}$$

3. High frequencies rejection noise

$$|T(j\omega)| = \frac{|C(j\omega)G(j\omega)|}{|C(j\omega)G(j\omega) + 1|} \leq \mathbf{A}dB, \omega \geq \omega_t \frac{rad}{s} \tag{2.11}$$

4. Output disturbances rejection

$$|S(j\omega)| = \frac{1}{|C(j\omega)G(j\omega) + 1|} \leq \mathbf{B}dB, \omega \leq \omega_s \frac{rad}{s} \tag{2.12}$$

5. Robustness to variations in the plant gain

$$\left.\frac{d[arg(C(j\omega)G(j\omega))]}{d\omega}\right|_{\omega=\omega_{cg}} = 0 \tag{2.13}$$

The solution of the simultaneous equations corresponding to this set of conditions is not a trivial problem, non-linear optimization techniques are used, which can be handled using computational tools.

2.5.2 Radial-based neural networks

Within the control paradigms we can find cascade control, feedforward control, model following among others and, in intelligent control there are genetic algorithms, particle swarm optimization, fuzzy control and neural networks. Genetic algorithms (GA) as well as particle swarm optimization (PSO) are similar in the sense that they look for minimal of a function that can have several local minimums. Artificial neural networks have in common with fuzzy logic that they are nonlinear methods to approximate functions [27].

A neural network with radial base function node (RBF) is conceptually very simple but an intrinsically powerful structure. A function is radially symmetric if its output depends on the distance of the input sample to convert the vector pattern into a scalar. Then it figures global approximations to functions using function combinations centered on weight vectors. It has been shown that RBF networks are universal estimators, so an RBF network with a sufficiently large number of nodes can approximate any continuous function of multiple variables into a compact set of data [1]. They can be applied efficiently to the domains of discrimination problems (such as voice verification), time series prediction (such as modeling in economics), and the extraction of features or even topographical maps. Figure 2.4 shows some examples of radial basis functions in the space of one dimension.

Figure 2.5 shows the architecture of the neural network. The input layer is simply a fan layer and does not perform any processing. The second layer performs a non-linear

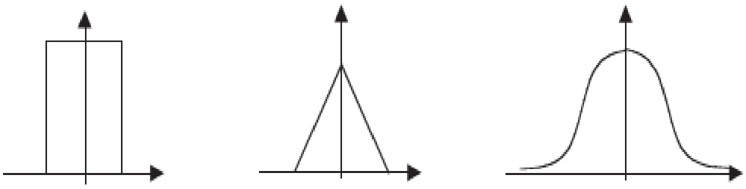


Figure 2.4: *Example of radial basis functions.*

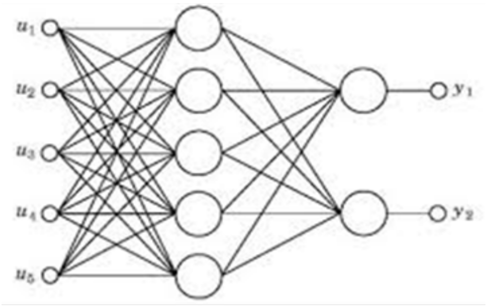


Figure 2.5: *Architecture of a neural network with a hidden layer.*

mapping from the input space to the output through the activation of the radial functions.

Several types of non-linear functions can be used, but the typical is Gaussian given the usefulness and because plenty of concepts that inspired these networks come from classic statistical techniques of pattern recognition [**?**, 49]. The following equation defines a radial basis function for each neuron of the intermediate, hidden layer

$$\varphi(||x - m||) = \exp[-\frac{||x - m||^2}{2\sigma^2}] \qquad 2.14$$

The number of hidden neurons affects how well the network is able to separate the data. A large number of hidden neurons will ensure correct learning, and the network is able to correctly predict the data with which it has been trained. With very few hidden neurons, the network may be unable to learn the relationships between the data and the error may be more than the suitable level. Therefore, selecting the number of hidden neurons is a crucial decision. With respect to the position of the centers or nodes, these can be kept fixed and equidistant or randomly placed.

To facilitate the modeling and design of fractional controllers it has been developed a set of tools or complementary tools (toolboxes) which help to simplify the necessary calculations. Three are representative for these effects, the first is CRONE [50] which includes design strategies through MATLAB and Simulink [51], the second is Ninteger [52] that helps in development and performance evaluation with MATLAB programs and the third is FOMCON [38, 53] which is also a toolbox for MATLAB; FOPID pursues the same goal but has not been yet highlighted by its impact or utility.

III

Analog design of a fractional PID controller

Chapter 3: Analog design of a fractional PID controller

3.1 Introduction

Proportional-Integral-Derivative (*PID*) controllers are the control strategies most used in today's industry. As shown in Fig. 2.2(a), this control strategy applies three actions (proportional, integral and derivative, with gains P, I and D, respectively) to a signal error $e(t)$=$r(t)$−$c(t)$ between the reference $r(t)$ and the signal being controlled $c(t)$. Typically, during the design stage the plant is modeled by its transfer function with integer orders q on the Laplace frequency s^q. However, experimental evidences show that physical systems can be modeled with higher accuracy using fractional order transfer functions [18], [20], [54]. On this direction, it is known that a fractional order $PI^{\lambda}D^{\mu}$ controller has better performance than the integer ones [18], [55], [56]. That is due to the addition of two degrees of freedom: one given by the integral action $\lambda \in \mathbb{R}^+$, and the second by the derivative action $\mu \in \mathbb{R}^+$ [18]. This allows us to include up to five design restrictions to the system response, namely: phase margin, gain margin, rejection to perturbations at the output, high-frequency noise rejection and robustness to variations in the gain of the plant [1], [57], [58], [38]. However, although fractional calculus has been studied from Leibniz in 1965, its practical use has been restricted.

It was not until the development of new computing environments and numerical calculus (e.g. MATLAB) when researchers introduced this theory to the modeling and control of systems [18], [38], [53], [43]. That way, in 1999 Podlubny proposed the first fractional controller [5]. Up to now one can find several realizations [54]- [36]. In addition, researchers have developed rules for parameters' syntonization, some of them considering Ziegler-Nichols rules [11], [59], [13] optimization methods [53], [19], techniques in the frequency domain [11], [60], [61], which offer robustness to the controller facing parametric uncertainties of the process and presence of external perturbations [57]; or also in techniques for intelligent computing, such as: neural networks [1], genetic algorithms [62], or fuzzy logic [41], [40]. In general, these techniques can be classified as analytical, numeric or rules-based ones. A summary of them is given in [63], [19], [36]. Unfortunately, the fractional order *PID* controller has not been appreciated by the industry as it is in the academia [18]. This is due to the implicit difficulties on the implementation in either digital or analog domains. In the analog case, the difficulties are accomplishing the design of circuit elements with frequency responses of the form $s^{-\lambda}$ or s^{μ} that are named “fractances".

The fractances are circuit elements with constant phase response at all frequencies [7]. For instance, very few physical realizations have been reported related to “fractal capacitances" [25], [23]. Unfortunately, those elements are bulky, require chemical compounds with difficult manipulation and the order λ cannot be modified easily. As alternatives, there exist approximations with rational functions in s for the operators $s^{-\lambda}$ or s^{μ}, that are obtained from Carlson methods, Matsuda, Oustaloup and from the continuous fractions expansion (CFE) [18], [6]. The resulting functions are implemented with arrays of resistances, capacitors and inductors in ladder networks of Cauer, Foster, and others [6]. The drawback of these realizations is the difficulty to approximate the required values with the ones of the commercially-available resistances and capacitors [26], in addition they can require negative impedance converters [6], [22] or inductors [64]. Another important issue is that digital $PI^{\lambda}D^{\mu}$ controllers [55] are not an option for controlling processes that change abruptly, as the case of vibrations [6].

From the difficulties on the implementations mentioned above, we propose two alternatives for the circuital realization of analog approximations for fractional derivatives and integrals, with the main advantage of using integrators of integer order, differential amplifiers, two-inputs adder amplifier, or from lead-lag networks. Most important is that the resulting circuits can be easily implemented with commercially-available resistances and capacitors and avoiding the use of negative impedance converters or inductors. In addition, the orders of the fractional derivative and integral can be modified just varying the gain of the differential amplifiers and adders, thus reducing design complexity. To validate the proposals we present simulation results using MATLAB and HSPICE, of a system with controllers Proportional-Integral (*PI*) and *PID*, both of fractional order, and a plant modeled as a system of order one plus dead time. The system is verified experimentally from a realization using operational amplifiers (OpAmps) uA741 and using an Application Specific Integrated Circuit (ASIC) that is known as Field-programmable Analog Array (FPAA) AN231E04 from Anadigm [65]. In this manner, the manuscript is organized as follows: Sect. 2 details the theory and introduces the concepts of fractional order derivative and integral, fractional order transfer function and the fractional order *PID* controller. Section 3 introduces our proposal as two alternatives for implementing the fractional derivative and integral. Section 4 describes the design of a system with *PI* and *PID* controllers of fractional order and two realizations are given by using OpAmps and an FPAA. Section 5 compares simulation and experimental results with a realization using Cauer networks. Finally, Sect. 3.5 lists the conclusions.

3.2 Proposal of implementation of fractional operators of Laplace

As mentioned before, the operators s^{a} and s^{-a} cannot be implemented directly, it is required to perform approximations. For instance, the approximation of order one with $a \in (0,1)$ obtained by CFE method has the form [7]

$$\frac{1}{s^{a}} \approx \frac{(1-a)s+(1+a)}{(1+a)s+(1-a)} = \frac{As+1}{s+A}, \qquad A = \frac{1-a}{1+a} \tag{3.1}$$

$$s^{a} \approx \frac{(1+a)s+(1-a)}{(1-a)s+(1+a)} = \frac{Bs+1}{s+B}, \qquad B = \frac{1+a}{1-a} \tag{3.2}$$

while the approximation of order two results in [7]

$$\begin{aligned} \frac{1}{s^{a}} &\approx \frac{(a^{2}-3a+2)s^{2}+(8-2a^{2})s+(a^{2}+3a+2)}{(a^{2}+3a+2)s^{2}+(8-2a^{2})s+(a^{2}-3a+2)} \\ &= \frac{Ds^{2}+Cs+1}{s^{2}+Cs+D} \end{aligned} \tag{3.3}$$

$$\begin{aligned} s^{a} &\approx \frac{(a^2+3a+2)s^2+(8-2a^2)s+(a^2-3a+2)}{(a^2-3a+2)s^2+(8-2a^2)s+(a^2+3a+2)} \\ &= \frac{Fs^2+Es+1}{s^2+Es+F} \end{aligned} \tag{3.4}$$

with

$$\begin{aligned} C &= \frac{8-2a^2}{a^2+3a+2}, \quad D = \frac{a^2-3a+2}{a^2+3a+2} \\ E &= \frac{8-2a^2}{a^2-3a+2}, \quad F = \frac{a^2+3a+2}{a^2-3a+2} \end{aligned} \tag{3.5}$$

From (3.1)-(3.5) we introduce two proposals of implementation for the operators s^{a} and s^{-a}.

3.2.1 Proposal of implementation 1

In this approach, the operators s^{a} and s^{-a} are implemented with integrators of integer order from CFE approximations of orders one and two. Besides, it can be generalized for approximations of higher order. Let (3.2) be the approximation of order one of the fractional derivative. The following transfer function is given

$$\frac{V_o(s)}{V_i(s)} = s^{a} \approx \frac{Bs+1}{s+B} \tag{3.6}$$

where V_i and V_o are the input and output voltages of the fractional derivative. Algebraic manipulation on (3.6) leads to

$$V_o(s)(s+B) = V_i(s)(Bs+1) \tag{3.7}$$

and after dividing both sides of (3.7) by s and regrouping similar terms, it results in

$$V_o(s) = \frac{V_i(s)-BV_o(s)}{s} + BV_i(s) \tag{3.8}$$

According to (3.8), the fractional derivative can be implemented from the block diagram shown in Fig. 3.1(a), by selecting gain blocks B. The realization using OpAmps is shown in Fig. 3.1(b). Circuit analysis to this circuit corresponds to the transfer function given by (3.6). The required design blocks are: differential amplifier with pondered gains (ADP) of value 1 and B, an adder with pondered gain (ASP), also of values 1 and B, and an inverter integrator (IInv). To tune the design, C_h is used as degree of freedom and it is chosen R=C_h^{-1}, while resistors of value $B^{-1}R$ are calculated with (3.2) for the desired order a. It is also possible to perform frequency denormalization by selecting a new value for C_h, equal to $C_h Omega$ to visit cut-off frequencies in (3.6) by a factor Ω.

Following these design procedures, this subsection introduces the implementation of

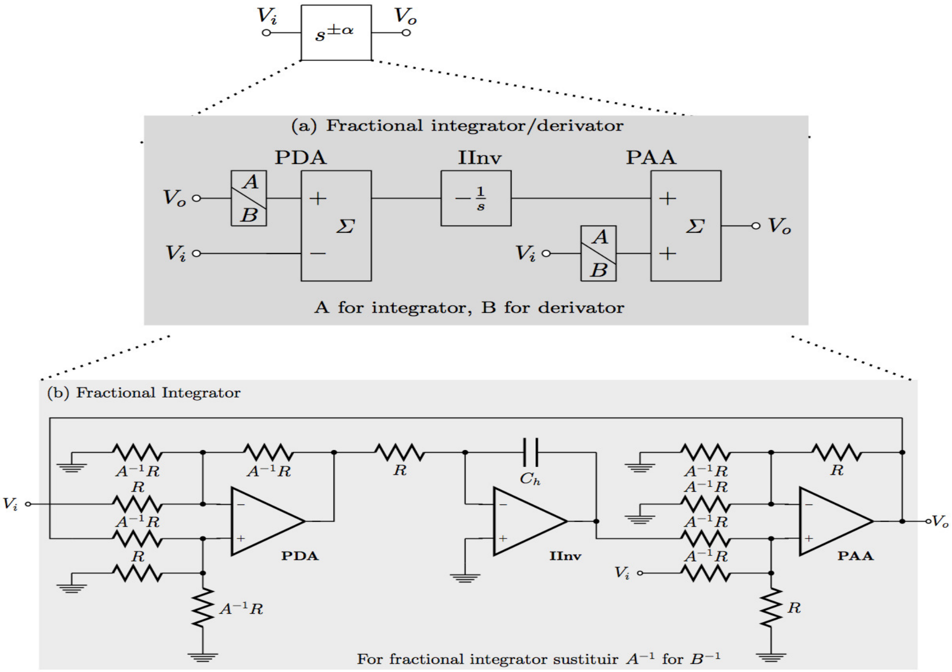


Figure 3.1: *(a) Block diagram of the proposal of implementation 1 of the fractional order derivator and integrator with a first order approximation; (b) Circuit realization.*

the fractional order integrator from (3.1) and the transfer function

$$\frac{V_o(s)}{V_i(s)} = \frac{1}{s^a} \approx \frac{As+1}{s+A} \tag{3.9}$$

Algebraic manipulation to (3.9) results in

$$V_o(s) = \frac{V_i(s) - AV_o(s)}{s} + AV_i(s) \tag{3.10}$$

and the fractional integrator is obtained from the block diagram shown in Fig. **??**(a), and selecting gain blocks A. The implementation using OpAmps is shown in Fig. **??**(b), the resistances have values $A^{-1}R$, with C_h as degree of freedom and R=C_h^{-1}. It is worth mentioning that the fractional integrator is directly obtained from the fractional derivator by substituting B by A in (3.2), choosing A in Fig. 3.1(a) and substituting B^{-1} by A^{-1} in Fig. 3.1(b). Again, a frequency denormalization can be performed by substituting C_h by C_hOmega. For the approximations of second order (3.3) and (3.4) the same procedure can be applied to obtain, respectively

$$V_o(s) = \frac{(V_i - V_oC)}{s} + \frac{(V_i - DV_o)}{s^2} + DV_i \tag{3.11}$$

$$V_o(s) = \frac{(V_i - V_oE)}{s} + \frac{(V_i - FV_o)}{s^2} + FV_i \tag{3.12}$$

these expressions can be realized with circuit elements from the block diagram shown in Fig. **??**. This procedure can be generalized for any approximation of order n of s^a and s^{-a}. A detailed study for selecting approximations of different order can be found in [7].

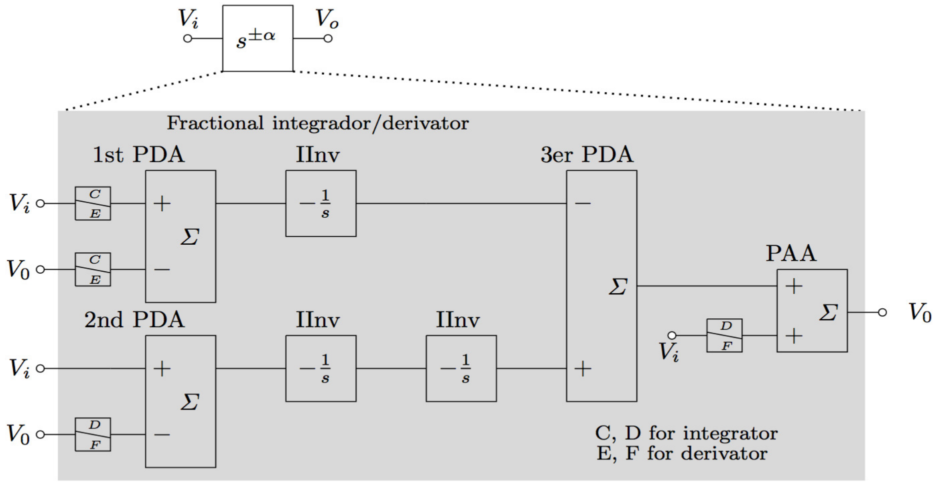


Figure 3.2: *Block diagram of the proposal of implementation 1 of the fractional order derivator and integrator with a second order approximation.*

3.2.2 Proposal of implementation 2

It is possible to obtain a more compact realization than that shown in Fig. 3.1(b), for approximations of order one of s^{a} and s^{-a}. To do that, this subsection proposes performing an adequate selection of the capacitors and resistances of the lead-lag network in Fig. 3.3. Circuit analysis is applied to obtain the transfer functions given by (3.1) and (3.2). Letting C_h as degree of freedom one can evaluate resistance values from B and A. A frequency denormalization can be performed by substituting C_h by $C_h Omega$.

3.3 Analog realization of the $PI^{\lambda}D^{\mu}$ controller

To validate the proposed circuit realizations for the Laplace operators of fractional order, this section reproduces the $PI^{\lambda}D^{\mu}$ controller and the system of order one plus dead time given in [1]. The plant $G(s)$ is modeled by

$$\begin{aligned} G(s) &= \frac{K}{1+sT}e^{-Ls} \approx \left(\frac{K}{1+sT}\right)\left(\frac{1-\dfrac{L}{2}s}{1+\dfrac{L}{2}s}\right) \\ &= \frac{K}{1+sT}e^{-0.1s} \approx \left(\frac{K}{1+sT}\right)\left(\frac{1-0.05s}{1+0.05s}\right) \end{aligned} \tag{3.13}$$

where T=1, L=0.1 y K={0.125, 0.25, 0.5, 1, 2, 4, 8} to evaluate the effect of variations in the gain of the plant. The dead time is modeled with an all-pass filter. The $PI^{\lambda}D^{\mu}$ controller has the form

$$\begin{aligned} C(s) &= P + Is^{\lambda} + Ds^{\mu} \\ &= 0.7469 + 0.874s^{1.2089} + 0.0001s^{0.0603} \end{aligned} \tag{3.14}$$

where P=0.7469, I=0.874, D=0.0001, λ=1.2089 and μ=0.0603 were computed in [**?**] to

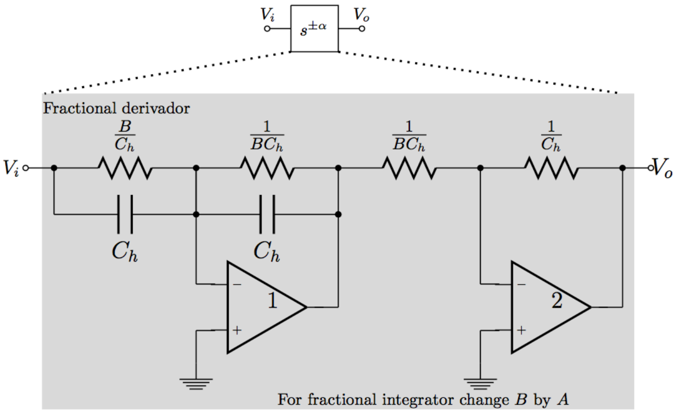


Figure 3.3: *Proposal of implementation 2 of the fractional order derivator and integrator with a first order approximation.*

satisfy the following design constraints:

- Zero crossing frequency ω_{cg}=0.5rad/s with

$$\left|C(j\omega_{cg})G(j\omega_{cg})\right|_{dB} = 0dB \tag{3.15}$$

- Phase margin φ_m=38.2^o where

$$arg\left(C(j\omega_{cg})G(j\omega_{cg})\right) = -\pi + \varphi_m \tag{3.16}$$

- High-frequency noise rejection A_x=$-10dB$ for frequencies $\omega \geq \omega_t$=10rad/s, where

$$\begin{aligned} \left|T(j\omega) = \frac{C(j\omega_{cg})G(j\omega_{cg})}{1 + C(j\omega_{cg})G(j\omega_{cg})}\right|_{dB} &\leq A_x dB \\ \forall\omega \geq \omega_t \quad \Rightarrow \quad |T(j\omega_t)|_{dB} &= A_x dB \end{aligned} \tag{3.17}$$

- Rejection to perturbations at the output B_x=$-20dB$ for frequencies $\omega \leq \omega_s$=0.01rad/s, where

$$\begin{aligned} \left|S(j\omega) = \frac{1}{1 + C(j\omega_{cg})G(j\omega_{cg})}\right|_{dB} &\leq B_x dB \\ \forall\omega \leq \omega_s \quad \Rightarrow \quad |S(j\omega_t)|_{dB} &= B_x dB \end{aligned} \tag{3.18}$$

- Robustness to variations in the gain of the plant.

$$\left.\frac{d\left[arg\left\{C(j\omega_{cg})G(j\omega_{cg})\right\}\right]}{d\omega}\right|_{\omega=\omega_{cg}} = 0 \tag{3.19}$$

3.3.1 Realization with the proposal of implementation 2

Figure 3.4 shows the circuit realization of the closed-loop system given in Fig. 2.2(b), with plant (3.13) and control (3.14), this last designed with the lead-lag network in Fig. 3.3

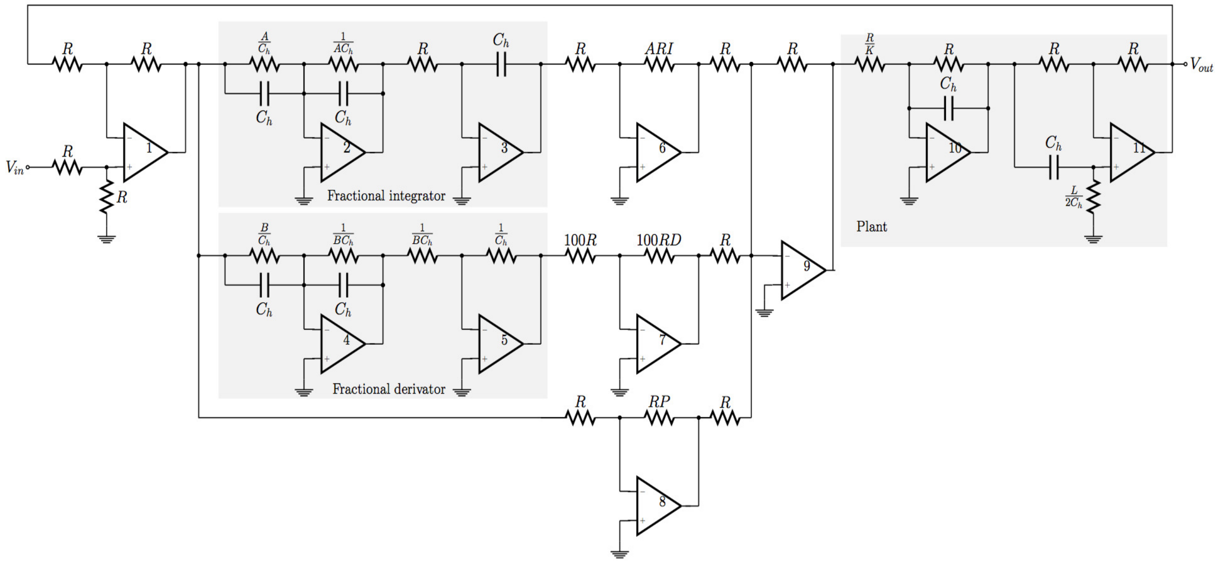


Figure 3.4: *Plant and $PI^{\lambda}D^{\mu}$ controller designed with the proposal of implementation 2.*

to approximate operators s^{λ} and $s^{-\mu}$. The controller requires 8 OpAmps, 5 capacitors and 17 resistances. To model the plant and comparator it requires 3 OpAmps, 2 capacitors and 9 resistances. The transfer function and the details of the design of every functional block of the circuit are listed in Table 3.1. The OpAmp labeled 1 and its four resistances R correspond to the comparator of the reference signal $r(t)=V_{in}(t)$ and the signal being controlled $c(t)=V_{out}(t)$. The output of this OpAmp is the error $e(t)=V_{in}(t)-V_{out}(t)$. The OpAmps labeled 2 and 3 form the fractional order integrator. Note that with respect to the circuit in Fig. 3.3, the resistance of value $1/C_h$ has been substituted by a capacitor of value C_h to incorporate an additional integrator of order one. This is due to the fact that λ=1.2089 and the approximation (3.1) is only valid for $\lambda \in (0,1)$. That way, the fractional integrator associated to OpAmp 2 is designed to $\hat{\lambda}=\lambda-1=0.2089$. On the other hand, the fractional derivator is realized with OpAmps 4 and 5, as shown in Fig. 3.3. The OpAmps 6, 7 and 8 in inverter connection set the integral gain I, derivative gain D, and proportional gain P, respectively. In addition, OpAmp 6 adds a gain factor A after substituting the resistance $1/C_h$ by the capacitor value C_h in the fractional integrator. The OpAmp 9 sum the three control actions and apply them to the plant, OpAmp 10 models a low-pass filter in series with an all-pass filter implemented with OpAmp 11. The design methodology is summarized as follows:

1. Choose the degree of freedom C_h=0.1mF and evaluate R=C_h^{-1}=10$K\Omega$.

2. Evaluate A=0.6543 and B=1.1283 from (3.1) and (3.2) for λ=0.2089 and μ=0.0603, respectively.

3. Given A, B, C_h, R, P, I, D, L and K, evaluate the remaining resistances from the values in Fig. 3.4.

4. The resistances with non-commercial values are approximated with series or parallel connections.

Table 3.1: *Design details of the* $PI^{\lambda}D^{\mu}$ *control of Fig.* 3.4 *with* C_h=0.1*mF and* Ω=6283.2.

Amp	Operation	Transfer function	Element	Theoretical value	Employed value
1	Comparator	V_o=V_{in}−V_{out}	R	10KΩ	10KΩ
2	Fractional integrator	$-\frac{1}{A}\left(\frac{As+1}{s+A}\right)$	A/C_h	6.54KΩ	6.6KΩ*
			$1/(AC_h)$	15.29KΩ	15.33KΩ*
			C_h/Ω	15nF	15nF
3	Integer integrator	$-\frac{1}{s}$	R	10KΩ	10KΩ
			C_h/Ω	15nF	15nF
4	Fractional derivator	$-\frac{1}{B}\left(\frac{Bs+1}{s+B}\right)$	B/C_h	11.28KΩ	11.2KΩ*
			$1/(BC_h)$	8.86KΩ	8.9KΩ*
			C_h/Ω	15nF	15nF
5	Inverter amplifier	B	$1/(BC_h)$	8.86KΩ	8.9KΩ*
			$1/C_h$	10KΩ	10KΩ
6	Integral gain	AI	R	10KΩ	10KΩ
			ARI	5.75KΩ	5.75KΩ*
7	Derivative gain	D	$100R$	1MΩ	1MΩ
			$100RD$	100Ω	100Ω
8	Proportional gain	P	R	10KΩ	10KΩ
			RP	7.46KΩ	7.4KΩ
9	Adder	V_o=V_1+V_2+V_3	R	10KΩ	10KΩ
10	Plant (lowpass filter)	$\frac{K}{1+sT}$	R	10KΩ	10KΩ
			R/K	$\frac{10K\Omega}{K}$	***
			C_h/Ω	15nF	15nF
11	Plant (allpass filter)	$\frac{1-\frac{L}{2}s}{1+\frac{L}{2}s}$	R	10KΩ	10KΩ
			$L/(2C_h)$	500Ω	500Ω**
			C_h/Ω	15nF	15nF

* Two resistors connected in series. ** Two resistors connected in parallel.

*** Array of resistors of 10KΩ

5. A frequency denormalization can be performed. For reducing the size of the capacitors to facilitate experimental measurements, one chooses Ω=6283.2, resulting in C_h=15*nF*.

3.3.2 Realization of the proposal of implementation 1

Figure 3.5 shows the circuit for realizing the system in Fig. 2.2(b), with plant (3.13), control (3.14), and with operators s^{λ} and $s^{-\mu}$, implemented from Fig. 3.1(b). With respect to the circuit in Fig. 3.4, only both the fractional integrator and derivator were modified, so that all the elements associated to OpAmps 1, 7, 8, 9, 10 and 11 have the same values listed in Table 3.1.

The transfer functions of all blocks of this circuit and design details are listed in Tables 3.1 and 3.2. Again, an integrator was added to fullfill λ=1.2089. In this case the

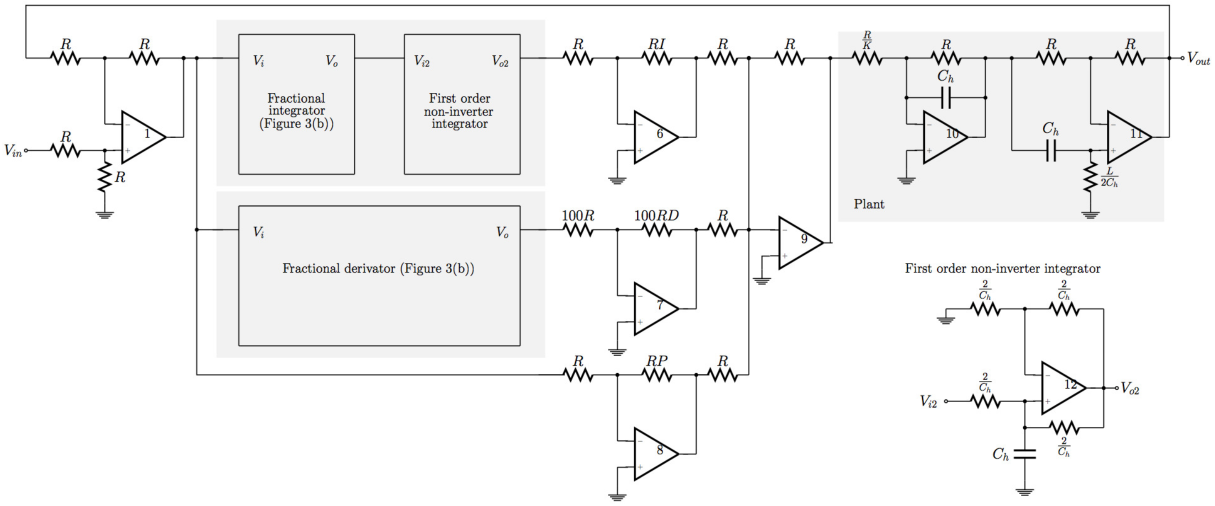


Figure 3.5: *Plant and $PI^{\hat{\lambda}}D^{\mu}$ controller designed with the proposal of implementation 1 of first order.*

Table 3.2: *Design details of the $PI^{\hat{\lambda}}D^{\mu}$ control of Fig. 3.5 with C_h=0.1mF and Ω=6283.2.*

Operation	Transfer function	Element	Theoretical value	Employed value
Fractional integrator	$\frac{As+1}{s+A}$	$1/(AR)$	15.27KΩ	15.27KΩ*
		R	10KΩ	10KΩ
		C_h/Ω	15nF	15nF
Non-inverter integrator	$\frac{1}{s}$	$2/C_h$	20KΩ	20KΩ*
		C_h/Ω	15nF	15nF
Fractional derivator	$\frac{Bs+1}{s+B}$	$1/(BR)$	8.9KΩ	8.88KΩ*
		R	10KΩ	10KΩ
		C_h/Ω	15nF	15nF
A6	I	R	10KΩ	10KΩ
		RI	8.74KΩ	8.76KΩ*

* Two resistors connected in series.   A1, A7-A11 as in Table 3.1

fractional integrator was designed for $\hat{\lambda}$=0.2089, and a non-inverting integrator was added by OpAmp 12, capacitor C_h, and resistances of value $2/C_h$. To accomplish A=0.6543, B=1.1283 and C_h=0.1mF, the resistances of the fractional integrator and derivator were evaluated as in Fig. 3.1(b). In addition, frequency denormalization was performed with Ω=6283.2, obtaining C_h=15nF. Also, non-commercial resistances were approximated using combinations of commercial ones. The control requires 11 OpAmps, 3 capacitors and 40 resistances.

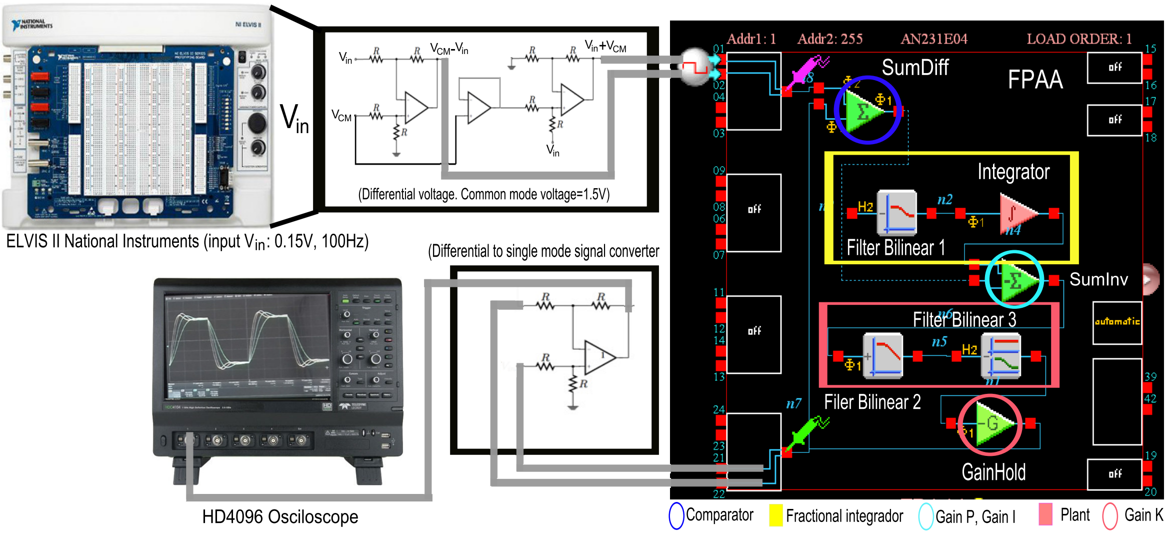


Figure 3.6: *Plant and PI$^{\lambda}$ controller designed in a FPAA and the corresponding experimental setup.*

3.3.3 Realization with Field Programmable Analog Arrays

Field Programmable Analog Arrays (FPAAs) are processors for analog signals, equivalents to the digital processors FPGAs (Field Programmable Gate Arrays). FPAAs are devices of specific purpose with the characteristics of being reconfigurable electrically. They are used to implement a variety of analog functions, such as: integration, derivation, pondered sum/subtraction, filtering, rectification, comparator, multiplication, division, analog-digital conversion, voltage references, signal conditioning, amplification, nonlinear functions, generation of arbitrary signals, among others. Since FPAAs are reconfigurable, one can implement complex prototypes in a short time. In this work the FPAA AN231E04 from Anadigm [**?**], is used. It uses technology of switched capacitors and it is organized into four configurable analog blocks (CABs). Those CABs are distributed in a matrix of size 2x2, supported by resources of programmable interconnections, seven configurable analog cells of input-output with active elements for amplification and dynamic reduction of offset and noise, an on-chip generator of multiple non-overlapped clock-signals and internal voltage references to eliminate temperature effects. It also includes a look-up table (LTU) of 8x256 bits for function synthesis and nonlinear signals, and for analog-digital conversion. The configuration data is saved into an internal SRAM, which allows reprogramming the device without interrupting its operation. The circuits are designed using the software Anadigmdesigner2, in which the user has access to a library of functional circuits CAMs (Configurable Analog Modules). Those CAMs are mapped in a portion for each CAB. The CABs has matrices of switches and capacitors, two OpAmps, a comparator, and digital logic for programming.

Figure 3.6 shows an implementation using FPAA AN231E04, equivalet to the circuit in Fig. 3.4, but avoiding the fractional derivator and its gain (OpAmp 7). Then, it uses the same plant and now controlled by a fractional *PI*. As shown in the following section, that control presents an almost identical performance to the fractional *PID* due to the fact that

Table 3.3: *Details of the design in FPAA of a PI$^{\eta}$ control with* Ω=6283.2.

Operation	CAM	Transfer function	Characteristics
Comparator	SumDiff (Sum/Difference)	$V_o = V_{in} - V_{out}$	G_1=1 G_2=-1
Fractional integrator	Filter Bilinear 1 (lag-lead function)	$-\frac{A\left(s+\frac{\Omega}{A}\right)}{s+A\Omega}$	$G=A$=0.655 $F_p=\frac{A\Omega}{2\pi}$=0.655KHz $F_z=\frac{\Omega}{2\pi A}$=1.526$KHz$
	Integrator	$\frac{\Omega}{s}$	$K_{int1}=\Omega$=0.006/μs
Gains P, I	SumInv (Inverter Adder)	$G_1V_1 - G_2V_2$	$G_1=-I$=−0.875 $G_2=-P$=−0.745
Plant	Filter Bilinear 2 (lowpass filter)	$\frac{\Omega}{s+\Omega}$	G=1 $F_c=\frac{\Omega}{2\pi T}$=1$KHz$
	Filter Bilinear 3 (allpass filter)	$\frac{\frac{2\Omega}{L}-s}{\frac{2\Omega}{L}+s}$	$F_c=\frac{2\Omega}{2\pi L}$=20$KHz$
	Gain Hold	$-K$	G=(0.25, 0.5, 1, 2)

G_i: gain at the i-th input of the CAM. F: corner frequency of the CAM
K_{int}: integration constant of the CAM

D=0.0001 and μ=0.0603 are low values compared to P=0.7469, I=0.874 and η=1.2089. Details of the design and the transfer functions of every block are listed in Table 3.3. The comparator producing the signal error $e(t)=V_{in}(t)-V_{out}(t)$ is implemented using a CAM “SumDiff" (adder-subtractor) configured with gains 1 and −1 at each input. The fractional integrator of order η=0.2089 is implemented by CAM “FilterBilinear 1" (Bilineal filter) and designed to produce a transfer function with one pole and zero given by (3.1). In series with this CAM, an “integrator" CAM is given to complete the order η=1.2089. The CAM “SumInv" sum the actions integral and proporcional, previously multiplied by P=0.7469 and I=0.874, according to (3.14). The plant is composed by a low-pass filter (CAM “Bilinear Filter 2"), an all-pass filter (CAM “Bilinear Filter 3"), and a gain block (CAM “Gain Hold") to adjust K. The frequency denormalization was performed with Ω=6283.2.

3.4 Results

To validate the proposals of implementation, this section presents both simulation and experimental results for the circuits in Fig. 3.4, Fig. 3.5, and Fig. 3.6. In addition, a

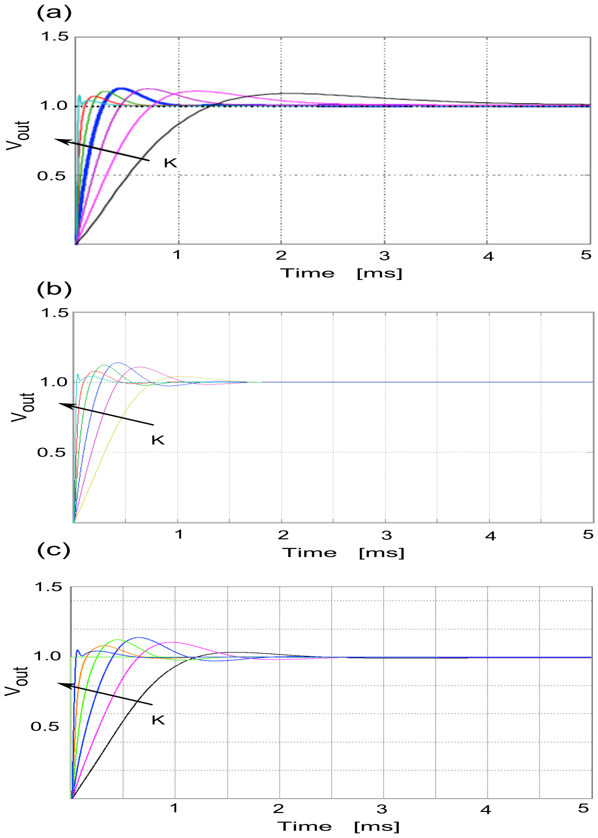


Figure 3.7: *(a) MATLAB simulation results reported in [**?**] for K=(0.125, 0.25, 0.5, 1, 2, 4, 8). (b) MATLAB/Simulink simulation of the system 3.13 with the $PI^{\lambda}D^{\mu}$ controller realized by means of the first implementation and K=(0.25, 0.5, 1, 2, 4, 8). (c) HSPICE simulation of the circuit of Fig. 3.5 with K=(0.25, 0.5, 1, 2, 4, 8).*

realization applying CFE technique is given for comparison purposes.

3.4.1 Simulink and HSPICE simulation with proposal of implementation 1

The system in Fig. 2.2(b) was simulated using Simulink with plant (3.13), control (3.14), Ω=6283.2 and the operators $s^{-\lambda}$ and s^{μ}, realized from Fig. **??**(a). The same system was simulated in HSPICE from Fig. **??**, for which the details of design are listed in Tables **??** and 3.2, and using the OpAmp uA741. Figure 3.7(a) reproduces the reported results in Fig. 3.2(a) in [**?**], while Fig. 3.7(b) and Fig. 3.7(c) show results by using MATLAB and HSPICE, respectively. Table **??** summarizes these results for K=(0.25, 0.5, 1, 2, 4, 8), an step-input of $1V$, offset=0.5V, and for a frequency of 100Hz. It can be observed that Fig. 3.7(b) and Fig. 3.7(c) show good agreement with the results given in [1], with exception of the cases when K=0.125 and K=0.25. This inconsistency is attributed to Padé approximation for the dead time rather that our proposal of implementation, in fact, the realization using Cauer networks shows a similar behavior. The remaining results demonstrate the validity of our proposal of implementation 1, with a maximum difference of 2.4% in the overshoot, and of 0.2ms in the settling time from 10% to 90%, with respect to the results given in [1] for K>0.5. An important issue is that, although the variations in K the overshoot remained narrow between 5.8% and 14% for K>0.5, thus confirming

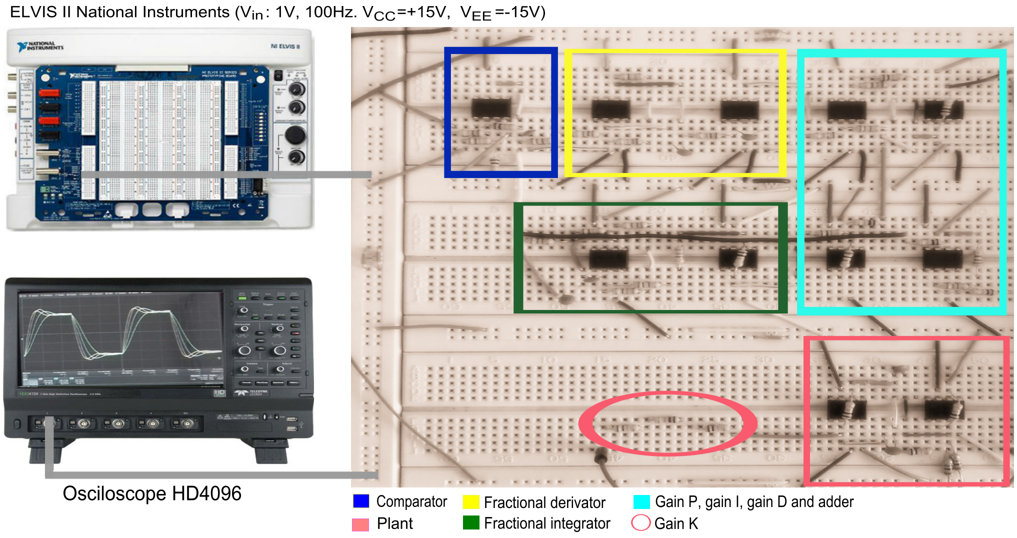

ELVIS II National Instruments (V_{in}: 1V, 100Hz. V_{CC}=+15V, V_{EE}=-15V)
Osciloscope HD4096
🟦 Comparator
🟨 Fractional derivator
🟦 Gain P, gain I, gain D and adder
🟥 Plant
🟩 Fractional integrator
○ Gain K

Figure 3.8: *Experimental setup of the system (3.13) with the $PI^{\lambda}D^{\mu}$ controller of Fig. 3.4.*

Table 3.4: *Results comparison*

	K=0.125		K=0.25		K=0.5		K=1		K=2		K=4		K=8	
Realization	M_p	t_s	M_p	t_s	M_p	t_s	M_p	t_s	M_p	t_s	M_p	t_s	M_p	t_s
Ref [**?**][1]	8%	1.46ms	11.2%	0.86ms	12.9%	0.52ms	14.23%	0.29ms	11.29%	0.17ms	7.3%	0.07ms	8.1%	0.021ms
Fig. 2.2(b) and 3.1(a)[2]	—	—	4.11%	0.54ms	10.9%	0.32ms	14%	0.2ms	12.3%	0.11ms	7.9%	0.052ms	5.8%	0.017ms
Fig. 3.5[3]	—	—	3.4%	0.82ms	10.5%	0.48ms	13.9%	0.29ms	12.3%	0.17ms	8%	0.081ms	6.1%	0.026ms
Fig. 3.5 and PI control[4]	—	—	3.4%	0.81ms	10.5%	0.48ms	13.8%	0.29ms	12.3%	0.16ms	8%	0.08ms	5%	0.026ms
Fig. 3.4[5]	4%	1.41ms	5.7%	0.99ms	9%	0.61ms	11.3%	0.36ms	10.5%	0.2ms	7.1%	0.094ms	8.2%	0.028ms
Fig. 3.6[6]	—	—	1.7%	0.46ms	11.4%	0.58ms	15%	0.431ms	10.4%	0.36ms	—	—	—	—
Fig. 3.12[7]	—	—	11.2%	0.82ms	13.46%	0.49ms	13.19%	0.29ms	10.9%	0.16ms	7.5%	0.082ms	4.5%	0.026ms

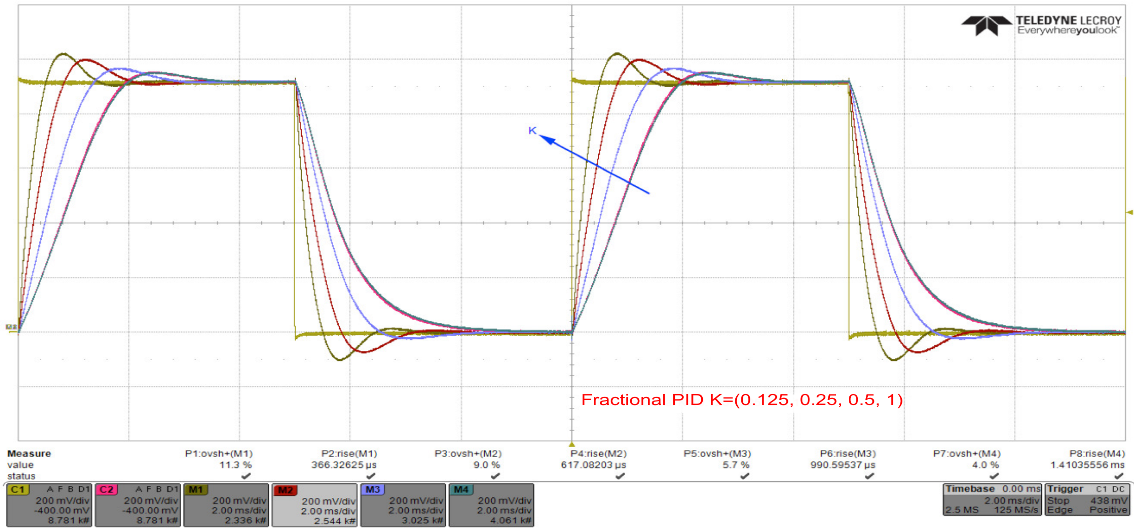
[1] Matlab simulation. [2] Matlab simulation, 1st implementation. [3] HSPICE simulation, 1st implementation. [4] HSPICE simulation, PI control.
[5] Experimental results, 2nd implementation. [6] Experimental results, 2nd implementation (FPAA). [7] HSPICE simulation.

the very low sensitivity to variations in K while satisfying (3.19).

3.4.2 Experimental validation with OpAmps and proposal of implementation 2

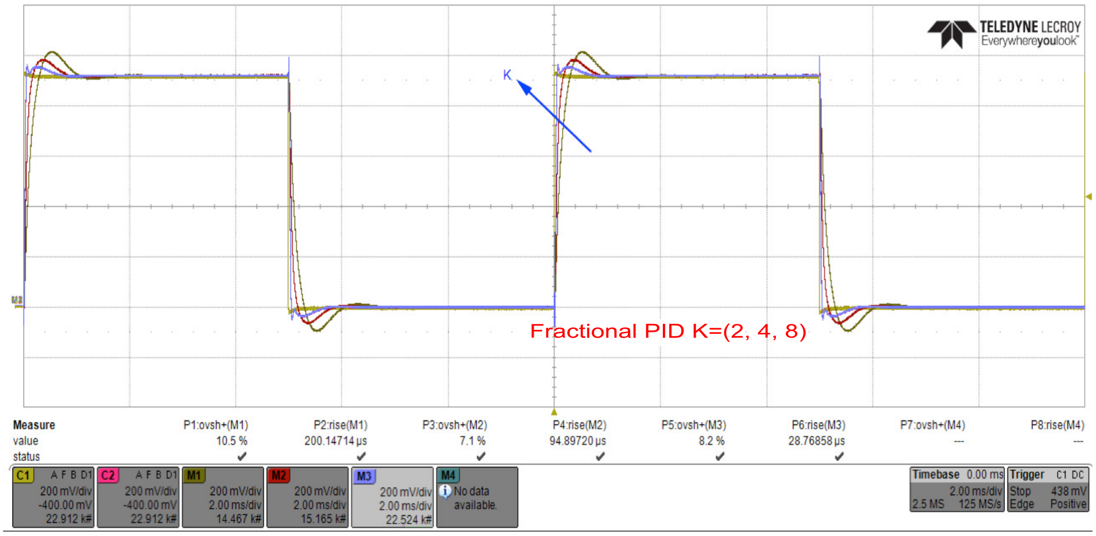
The system in Fig. 2.2(b) was implemented on protoboard with plant (3.13), control $PI^{\lambda}D^{\mu}$ (3.14), Ω=6283.2, and operators $s^{-\lambda}$ and s^{μ}, realized with OpAmps uA741 from Fig. 3.4 and the design details listed in Table 3.1. The experimental configuration is shown in Fig. 3.8, which consists of an input square signal of $1V$ in amplitude, offset=$0.5V$, and frequency of $100Hz$, supplied from the experimental platform ELVIS II from National Instruments. This device also provides bias voltages of $\pm 15V$ to the OpAmps. The output is measured with an Oscilloscope HD4096 Teledyne Lecroy. Figure 3.9 and Fig. 3.10 show the results for K=$(0.125, 0.25, 0.5, 1)$ and K=$(2, 4, 8)$, respectively. Table 3.4 summarizes these results. The gains K were adjusted with arrays of resistances of $10K\Omega$.

It can be noted that the simulations results in Fig. **??**, and the experimental ones


TELEDYNE LECROY
Everywhereyoulook
K
Fractional PID K=(0.125, 0.25, 0.5, 1)
C1 A F B D1 200 mV/div -400.00 mV 8.781 k#
C2 A F B D1 200 mV/div -400.00 mV 8.781 k#
M1 200 mV/div 2.00 ms/div 2.336 k#
M2 200 mV/div 2.00 ms/div 2.544 k#
M3 200 mV/div 2.00 ms/div 3.025 k#
M4 200 mV/div 2.00 ms/div 4.061 k#
Timebase 0.00 ms 2.00 ms/div 2.5 MS 125 MS/s
Trigger C1 DC Stop 438 mV Edge Positive

Measure	P1:ovsh+(M1)	P2:rise(M1)	P3:ovsh+(M2)	P4:rise(M2)	P5:ovsh+(M3)	P6:rise(M3)	P7:ovsh+(M4)	P8:rise(M4)
value	11.3 %	366.32625 µs	9.0 %	617.08203 µs	5.7 %	990.59537 µs	4.0 %	1.41035556 ms
status	✓	✓	✓	✓	✓	✓	✓	✓

Figure 3.9: *Experimental results with the second implementation for K=(0.125, 0.25, 0.5, 1).*


TELEDYNE LECROY
Everywhereyoulook
K
Fractional PID K=(2, 4, 8)
C1 A F B D1 200 mV/div -400.00 mV 22.912 k#
C2 A F B D1 200 mV/div -400.00 mV 22.912 k#
M1 200 mV/div 2.00 ms/div 14.467 k#
M2 200 mV/div 2.00 ms/div 15.165 k#
M3 200 mV/div 2.00 ms/div 22.524 k#
M4 No data available.
Timebase 0.00 ms 2.00 ms/div 2.5 MS 125 MS/s
Trigger C1 DC Stop 438 mV Edge Positive

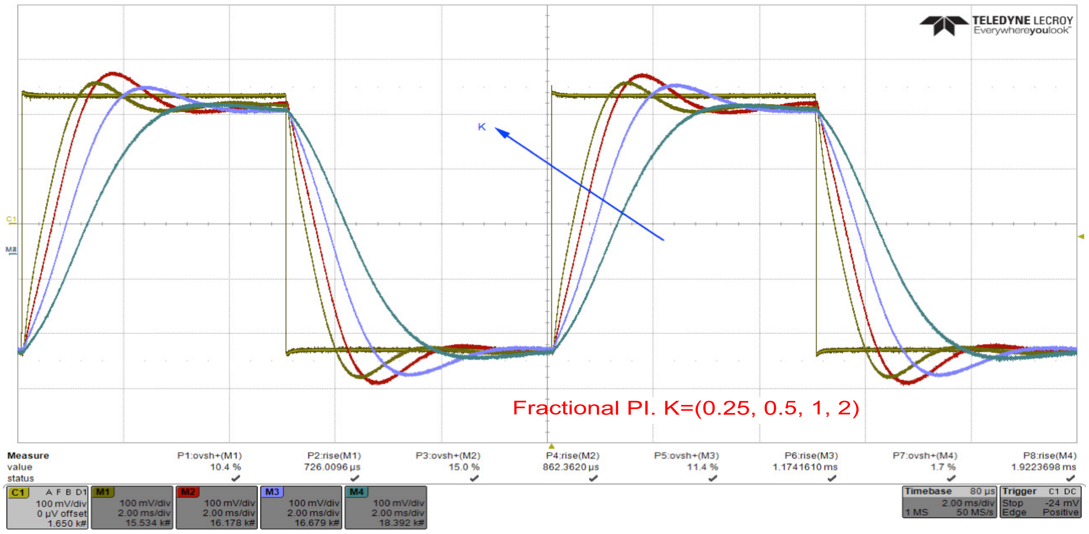
Measure	P1:ovsh+(M1)	P2:rise(M1)	P3:ovsh+(M2)	P4:rise(M2)	P5:ovsh+(M3)	P6:rise(M3)	P7:ovsh+(M4)	P8:rise(M4)
value	10.5 %	200.14714 µs	7.1 %	94.89720 µs	8.2 %	28.76858 µs	---	---
status	✓	✓	✓	✓	✓	✓		

Figure 3.10: *Experimental results with the second implementation for K=(2, 4, 8).*

in Fig. 3.9 and Fig. 3.10 are in good agreement, thus validating our proposal of implementation 2. Also, there exists agreement with the settling time from 10% to 90%, which is between 0.028ms and 1.41ms. The overshot behaves similar between 4% and 11%, demonstrating again that this design has very low sensitivity to variations in the gain K.

3.4.3 Experimental validation with FPAA AN231E04

When the circuit in Fig. 3.5 was simulated in HSPICE, it was investigated the effect of avoiding the effect of the derivative action. The corresponding results are summarized in Table 3.4. As one sees, the results are quite similar to that of the $PI^{\lambda}D^{\mu}$ controller. Therefore, the implementation of the fractional PI was done using the FPAA AN231E04, as described in Sect. 3.3.3. The experimental configuration of Fig. 3.6 has a differential input of 0.5V in amplitude and frequency 100Hz, and a common-mode component


TELEDYNE LECROY
Everywhereyoulook
K
C1
M8
Fractional PI. K=(0.25, 0.5, 1, 2)
C1 A F B D1 100 mV/div 0 µV offset 1.650 k#
M1 100 mV/div 2.00 ms/div 15.534 k#
M2 100 mV/div 2.00 ms/div 16.178 k#
M3 100 mV/div 2.00 ms/div 16.679 k#
M4 100 mV/div 2.00 ms/div 18.392 k#
Timebase 80 µs 2.00 ms/div 1 MS 50 MS/s
Trigger C1 DC Stop -24 mV Edge Positive

Measure	P1:ovsh+(M1)	P2:rise(M1)	P3:ovsh+(M2)	P4:rise(M2)	P5:ovsh+(M3)	P6:rise(M3)	P7:ovsh+(M4)	P8:rise(M4)
value	10.4 %	726.0096 µs	15.0 %	862.3620 µs	11.4 %	1.1741610 ms	1.7 %	1.9223698 ms
status	✓	✓	✓	✓	✓	✓	✓	✓

Figure 3.11: *Experimental results with the proposal implementation 2 in FPAA, for K=$(0.25, 0.5, 1, 2)$ (PI control).*

of $1.5V$ from the array of three OpAmps and target ELVIS II. The output is converted from differential mode to simple mode with a differential amplifier. To compare the obtained results with respect to previous realizations, the settling time was doubled to compensate the input signal of $0.5V$ instead of $1V$. This difference is imposed by the input voltage range that the FPAA can drive without modifying the transfer functions of the already designed blocks. The settling times of 10% to 90% are not accurate in this case but allow performing the comparison. In this manner, the measurements were performed with the Oscilloscope HD4096. Figure 3.11 and Table 3.4 summarize the results for K=$(0.25, 0.5, 1, 2)$. As one sees, there is again a good agreement with the ones for K>0.25, with an overshoot between 10.4% and 15%, and the rise time between $0.36ms$ and $0.58ms$. One can observe a gain error once the overshoot is attenuated, and this is attributed to the gain error of the blocks in the FPAA instead of the steady state, which must be zero due to the existence of integrators in the controller. Again the overshoot has very low sensitivity to variations in the gain of the plant.

3.4.4 Simulation with $PI^{\hat{\lambda}}D^{\mu}$ control from Cauer networks

For comparison purposes, the system in Fig. 2.2(b) was simulated in HSPICE with plant (3.13), control $PI^{\hat{\lambda}}D^{\mu}$ (3.14), Ω=6283.2, and operators $s^{-\hat{\lambda}}$ and s^{μ} substituted by the approximations of order two (3.3) and (3.4), and implemented applying the first Cauer method (Fig. 2.1). Figure 3.12 shows the resulting circuit, which was realized using the design details listed in Table 3.5. One can appreciate that with respect to the circuit in Fig. 3.4 only the fractional integrador and derivator were modified, so that all elements associated to OpAmps 1, 3, 7, 8, 9, 10 and 11 remain with the same values in Table **??**. In the same manner as for the circuit in Fig. 3.5, the inverter amplifier realized with OpAmp 6 is conserved but with a new resistance ARI substituted by RI. Figure 3.13 shows the HSPICE simulation with an amplitude input of $1V$, offset=$0.5V$, and frequency of $100Hz$.

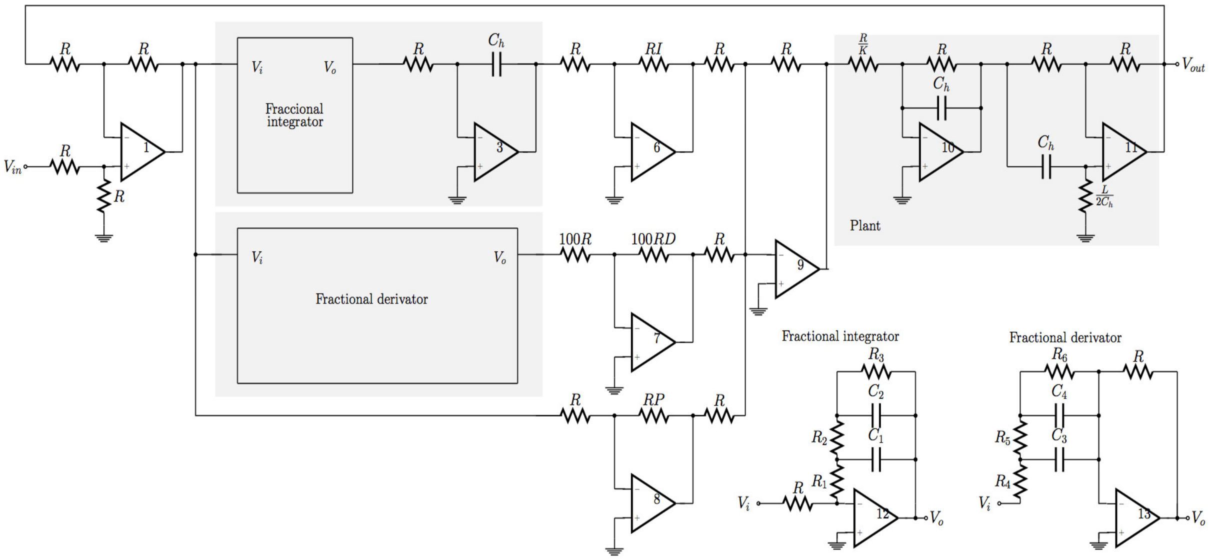


Figure 3.12: *Plant and control system $PI^{\lambda}D^{\mu}$ from continuous expansion of fractions and Cauer networks.*

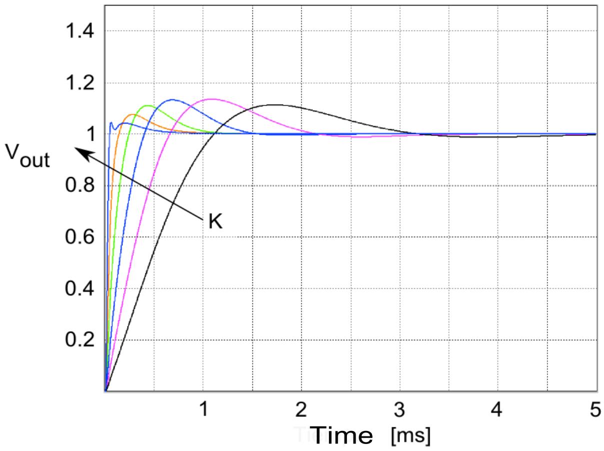


Figure 3.13: *HSPICE simulation of the circuit of Fig. 3.12 for* K=(0.25, 0.5, 1, 2, 4, 8).

The results are summarized in Table 3.4. Again, there exists good agreement with the results of the other circuits, and in this case the results for K=0.25 are better approached to the ones reported in [1]. The opposite occurs for K=8. In addition, this system looks quite compact and simple than the ones proposed in Fig. 3.4 and Fig. 3.5, in this case the fractional *PID* controller requires 7 OpAmps, 19 resistances and 5 capacitors.

3.4.5 Discussion

Table 3.6 presents a comparison among *PID* controllers in Figs. 3.4, 3.5 and 3.12, in terms of the number of active and passive elements, and in terms on the design complexity. It can be noted that the second proposal of implementation is the most compact since it requires the less number of OpAmps and resistances. It just requires 6 additional resistances to approximate the required ones with commercial values. It can be appreciated also that for being degrees of freedom, the capacitors can always be implemented with

Table 3.5: *Design details of the $PI^{\lambda}D^{\mu}$ control of Fig. 3.12 with C_h=0.1mF, Ω=6283.2 and a magnitude denormalization of 10,000 in the elements of the fractional integrator and fractional derivator.*

Operation	Transfer function	Elemen	Theoretical value	Employed value
Fractional integrator	$\frac{D_1 s^2 + C_1 s + 1}{s^2 + C_1 s + D_1}$	R	10$K\Omega$	10$K\Omega$
		R_1	5.3$K\Omega$	5.38$K\Omega^*$
		R_2	5.7$K\Omega$	5.7$K\Omega^*$
		R_3	8.33$K\Omega$	8.32$K\Omega^*$
		C_1	11.42nF	11nF*
		C_2	88.3nF	90nF*
Fractional derivator	$\frac{1}{\frac{D_2 s^2 + C_2 s + 1}{s^2 + C_2 s + D_2}}$	R	10$K\Omega$	10$K\Omega$
		R_1	8.35$K\Omega$	8.32$K\Omega^*$
		R_2	52.36$K\Omega$	51$K\Omega^*$
		R_3	$-24.5K\Omega$	requiere NIC
		C_1	11.42nF	11nF*
		C_2	-13nF	requiere NIC
A6 (integral gain)	I	R	10$K\Omega$	10$K\Omega$
		RI	8.74$K\Omega$	8.76$K\Omega^*$

*Series combination of two resistors or two capacitors.
A1, A7-A11 as in Table 3.1.

commercial values. The second proposal of implementation requires the same number of active elements as the one with Cauer networks, and only 3 capacitors, but the number of resistances is considerable. This is due to the fact that 20 resistances were added to approximate the required values. That number can be reduced to 12 by interchanging the resistances of value R and $A^{-1}R$ (or $B^{-1}R$) in Fig. 3.1(b), and if changing A^{-1} (B^{-1}) by A (B). On the other hand, the design of the controller with this proposal is more simple since one just needs to evaluate two resistances: $A^{-1}R$ and $B^{-1}R$. The design with Cauer networks is the most complicated one because it performs a continuous expansion of fractions, and if necessary, it requires negative impedance converters. Another difficulty is the lack of degrees of freedom, for which all resistances and capacitors will be of non-commercial values. Finally, although the comparisons among the proposed approximations are of order 1 and the design with Cauer networks is order 2, the results listed in Table 3.4 show that there is not a notably difference among the realizations in terms of the performance, thus concluding that the realizations of order 1 are pretty good enough to realize fractional *PID* controllers.

Table 3.6: *Results comparison*

	OpAmps			Resistors			Capacitors			NICs*	Dificultad	
	R	A	Total	R	A	Total	R	A	Total		Procedure	Complexity
1st proposal	8	0	**8**	17	6	**23**	5	0	**5**	**0**	Calculate A and B with (3.1) and (3.2) using λ and μ and substitute values in Figure 3.3	Low
2nd proposal	11	0	**11**	40	20 12	**60** **52**	3	0	**3**	**0**	Calculate A and B with (3.1) and (3.2) using λ and μ and substitute values in Figure 3.1(b)	Very low
Cauer	7	4	**11**	19	15	**34**	5	3	**8**	**2**	Calculate C, D, E and F with (3.5) using λ and μ, perform the CFE decomposition and find the equivalent Cauer network. If it is required design the required NICs	High
Ref [?] fractional PD	4	0	**4**	12	(12, 24)	**(24, 36)**	7	(7, 14)	**(14, 21)**	0	Seven equations must be solved to obtain the values of required resistors and capacitors. These elements are approximated with serial/parallel connections of two or three elements of values delivered in standard series	High

R: Number of required elements. A: Number of elements added because of NICs or to approximate non-commercial values of capacitance or resistance.
NIC: Negative Impedance Converter.

3.5 Conclusions

Two alternatives for the analog implementation of integrators and derivators of fractional order have been introduced. They include implementations with lead-lag networks or with differential amplifiers, adders and integrators of integer order. These proposals were validated simulating (MATLAB and HSPICE) *PI* and *PID* controllers of fractional order, and with their realizations using commercially available OpAmps and FPAAs. After comparing the results with respect to an equivalent realization from Cauer networks, it was appreciated that in addition to a reduction in the design process, also the number of active and passive elements being reduced, and they have the advantage of including degrees of freedom that allow selecting circuit elements or commercial values. These advantages could increase the use of fractional order *PID* controllers in industrial applications. Most important is that our proposed implementations for fractional order integrators and derivators can be extended to other applications like fractional order filters [66], chaotic systems [67], or fractional memristors [68].

IV

Fractional PID controller tuning

Chapter 4: Fractional PID controller tuning

4.1 Tuning by simultaneous non-linear optimization

The tuning of the five controller parameters K_p, K_i, λ, K_d and μ of the $PI^{\lambda}D^{\mu}$ controller presented in section 2.3 is directly associated to the plant parameters and the desired performance of the system. In the procedure proposed by *Monje et al* [9] five simultaneous non linear inequalities with restrictions are developed in terms of the variables of the controller (K_p, K_i, λ, K_d and μ), the plant (T and L) and the desired performance established by means of the phase and gain margin, noise and disturbance rejection as well as variations in the plant. It has been suggested in some publications [56, 69] to use the computational optimization function FMINCON, which is installed on software as MATLAB and SCILAB to perform the task of finding optimal values for each controller parameter of the $PI^{\lambda}D^{\mu}$ controller. It finds a constrained minimum of a scalar function of several variables starting at an initial estimate. In such function must be stated all features we have, including the target or main function, nonlinear conditions to be fulfilled in the form of equations or inequalities, a set of lower and upper bounds of the independent variables and the range of output values and initial conditions to optimize parameters [1]
.

Once it has been clearly defined the performance requirements and the plant model, it can be used the toolbox of MATLAB called FOMCON [38, 53]. FOMCON is the result of an interdisciplinary research project dedicated to the development and applications of fractional calculus, its distribution and installation procedure is available from the project page [70].

4.2 Design requirements of the system

Summarizing the conditions the system must meet, here is the 5 performance specifications: $GM = 20dB$, $PM = 51.83°$ if $\zeta = 0.5$ (or $MP \simeq 16.3\%$), $T(s) \leq -10dB$ $\forall \omega \geq \omega t = 10rad/s$, $S(s) \leq -20dB$ $\forall \omega \leq \omega s = 0.01rad/s$ and $\omega_{cg} = 0.5rad/s$.

1. Phase margin

$$|C(j\omega_{cg})G(j\omega_{cg})|_{\omega_{cg}=\frac{1}{2}\frac{rad}{s}} = 0dB \qquad (4.1)$$

2. Gain margin

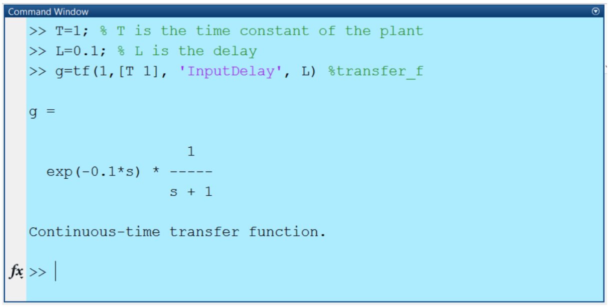
$$arg(C(j\omega_{cg})G(j\omega_{cg})) = -\pi + \frac{2}{3}\frac{rad}{s} \qquad (4.2)$$

3. High frequencies rejection noise

$$|T(j\omega)| = \frac{|C(j\omega)G(j\omega)|}{|C(j\omega)G(j\omega)+1|} \leq -10dB, \omega \geq 10\frac{rad}{s} \qquad (4.3)$$

4. Output disturbances rejection

$$|S(j\omega)| = \frac{1}{|C(j\omega)G(j\omega)+1|} \leq -20dB, \omega \leq \frac{1}{100}\frac{rad}{s} \qquad (4.4)$$



```
>> T=1; % T is the time constant of the plant
>> L=0.1; % L is the delay
>> g=tf(1,[T 1], 'InputDelay', L) %transfer_f

g =

                  1
  exp(-0.1*s) * -----
                s + 1

Continuous-time transfer function.

>> |
```

Figure 4.1: *MATLAB variables of a fractional transfer function registered in the workspace.*

5. Robustness to variations in the plant gain

$$\left.\frac{d[arg(C(j\omega)G(j\omega))]}{d\omega}\right|_{\omega=\omega_{cg}} = 0 \qquad (4.5)$$

4.3 Tuning through FOMCON

This section illustrates the use of dialog boxes of FOMCON's interface in order to tune the fractional PID controller. The tuning procedure is listed below [71] :

1. State the plant model.
2. Find the stability conditions of the plant (maximum loop gain and permitted delay).
3. Assign the maximum of loop gain value to the upper bound of K_p.
4. Set the proposed ranges of the remaining four parameters and a value to start.
5. Fix the performance conditions of the system.
6. Choose the optimization algorithm.
7. Run the application.
8. Check that the result meet the specifications

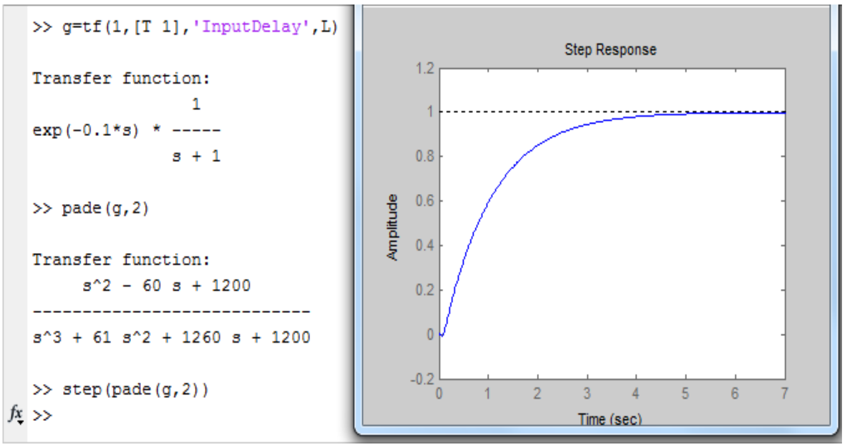
4.3.1 Model plant statement

Let the plant be modeled as a first order transfer function with delay or dead time. This transfer function is declared in MATLAB by the statements on the command window shown in Figure 4.1.

4.3.2 Stability conditions

The transfer function G(s) includes a pure delay. It can be written for analysis purposes by means of an *nth* approximation of Padé method for any desired order.

In MATLAB's command window is introduced *pade(g,2)*, the arguments are the function to approximate G(s) and the expansion order. In Figure 4.2 it is exemplified the



```
>> g=tf(1,[T 1],'InputDelay',L)

Transfer function:
                  1
exp(-0.1*s) * -----
              s + 1

>> pade(g,2)

Transfer function:
     s^2 - 60 s + 1200
----------------------------
s^3 + 61 s^2 + 1260 s + 1200

>> step(pade(g,2))
>>
```

Figure 4.2: *Padé approximation and its plot on MATLAB.*

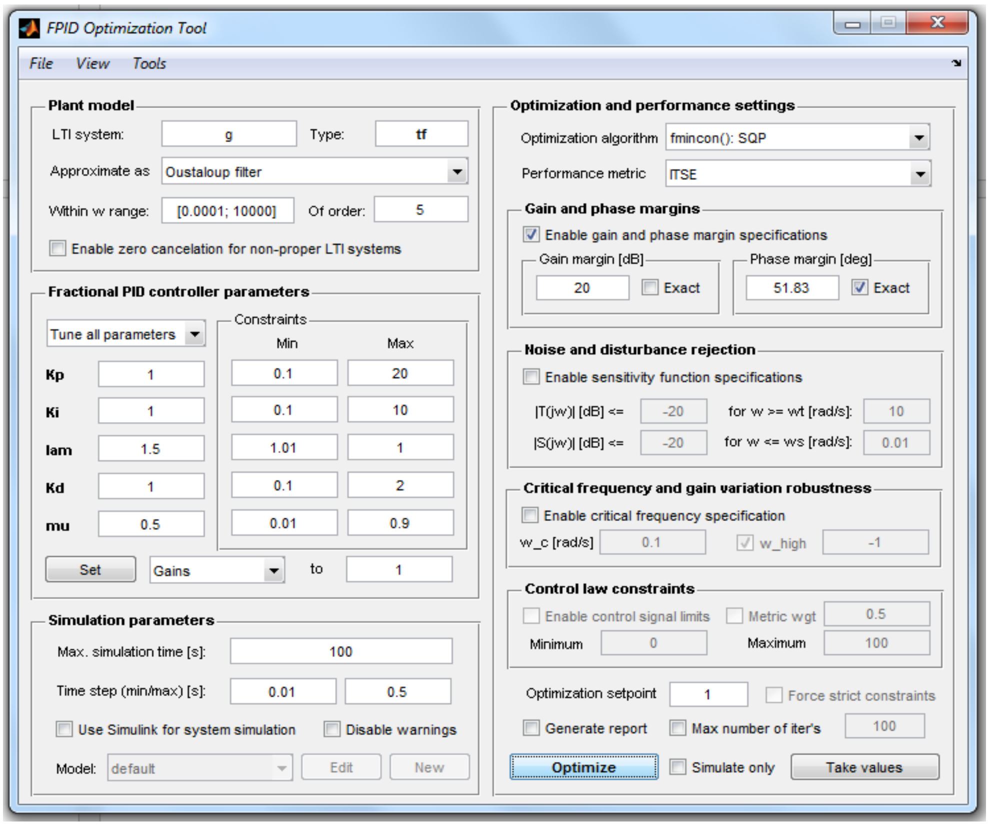
approach. To simplify the procedure of stability analysis can be used a first order approximation of the delay expression multiplied by the transfer function of the plant, then apply Routh-Hurwitz criterion [33] to determine the maximum L and K from T. In the example above stability is ensured if $K < (2 * T + L)/L$.

4.3.3 Initial conditions and settings

Once FOMCON files are in the root directory of MATLAB the command *fpid_optim* is typed and a new window pops up (Figure 4.3), that window contains four main sections. Plant features, controller parameters, simulation parameters, algorithm and frequency performance settings are inserted there. Some boxes are optional and other necessary to start the tuning process.

The box corresponding to **Plant model** allows to call the name of the plant present in the workspace, in this case it has been used g meaning the function G(s). The plant type is automatically assigned when the 'Optimize' button is pressed and there are two types, **tf** and **ftf**, ordinary or fractional transfer function, this identification depends on how G(s) was declared. The section **Fractional PID controller parameters** includes three columns. The first column is for placing the initial values of the five parameters K_p, K_i, λ, K_d and μ. It also has a tab with three tuning options: all parameters at once, stop fixed gains or fixed exponents. The other two columns let assign the constraints or bounds among optimization. In the case of K_p, K_i and K_d gains an initial value of 1 can be assigned, however the range of K_p as the main gain of the controller should be a number greater than zero but less than the maximum gain allowed for the plant stability in closed-loop, in this case it has been used the following values: $0.1 \leq K_p \leq 20$, $0.01 \leq K_i \leq 10$ and $0.01 \leq K_d \leq 10$; the powers $1.01 \leq \lambda \leq 2$ and $0.01 \leq \mu \leq 0.9$ with initial values at the midpoint of the range, 1.5 and 0.5 respectively. The box **Simulation parameters** is basically a section to adjust how long the simulation lasts and step-size, it could be left intact at the beginning.

The **Optimization and performance settings** allow to specify the algorithm and conditions to apply the controller tuning in the frequency domain. The toolbox comprises two



FPID Optimization Tool

Plant model

LTI system: g Type: tf

Approximate as: Oustaloup filter

Within w range: [0.0001; 10000] Of order: 5

□ Enable zero cancelation for non-proper LTI systems

Fractional PID controller parameters

Tune all parameters

		Constraints Min	Constraints Max
Kp	1	0.1	20
Ki	1	0.1	10
lam	1.5	1.01	1
Kd	1	0.1	2
mu	0.5	0.01	0.9

Set Gains to 1

Simulation parameters

Max. simulation time [s]: 100

Time step (min/max) [s]: 0.01 0.5

□ Use Simulink for system simulation □ Disable warnings

Model: default Edit New

Optimization and performance settings

Optimization algorithm: fmincon(): SQP

Performance metric: ITSE

Gain and phase margins

☑ Enable gain and phase margin specifications

Gain margin [dB]: 20 □ Exact

Phase margin [deg]: 51.83 ☑ Exact

Noise and disturbance rejection

□ Enable sensitivity function specifications

|T(jw)| [dB] <= -20 for w >= wt [rad/s]: 10

|S(jw)| [dB] <= -20 for w <= ws [rad/s]: 0.01

Critical frequency and gain variation robustness

□ Enable critical frequency specification

w_c [rad/s] 0.1 ☑ w_high -1

Control law constraints

□ Enable control signal limits □ Metric wgt 0.5

Minimum 0 Maximum 100

Optimization setpoint 1 □ Force strict constraints

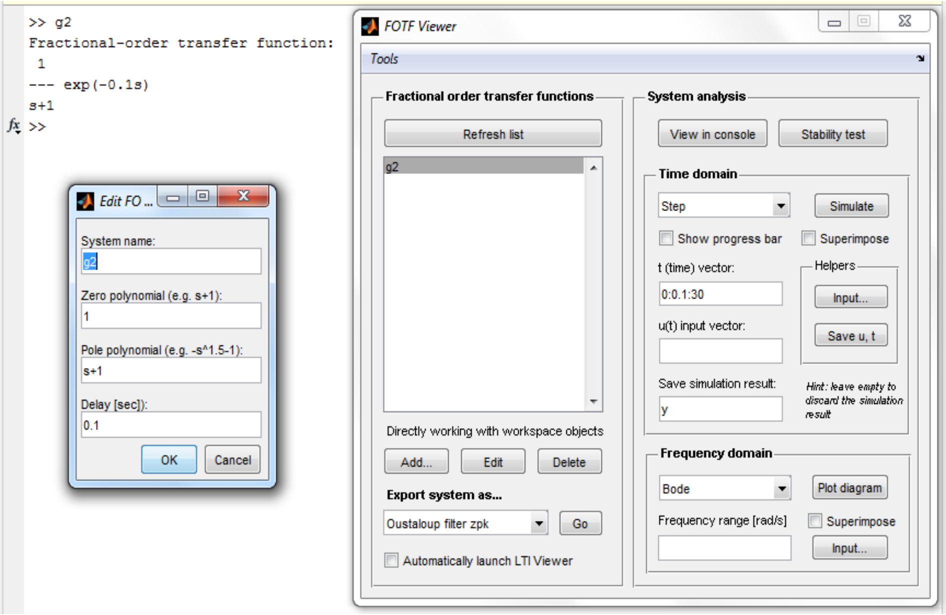
□ Generate report □ Max number of iter's 100

Optimize □ Simulate only Take values

Figure 4.3: *FOMCON tuning interface running in MATLAB.*

optimization algorithms and four options as performance metrics. Using the tabs on the upper right corner we select *optimize* or *fmincon* functions with stop criteria *ISE*, *IAE*, *ITAE* or *ITSE*; respectively: integral square error, integral absolute error, integral time-square error and integral time-absolute error. Each combination of the two options has different performance results, tests were performed with the same initial conditions and each simulation behaves differently. For example, a response with more overshoot than expected was using *optimize* and a solution with *fmincon* spent more time to converge or had more setting time instead because wider oscillations (*fmincon: IP with ISE*). The best option we found for solving the proposed systems was *fmincon: SQP with ITSE*, its execution is fast and both the time response and Bode plot are suitable.

Finally, three subsections allow us to enable the performance specifications that the whole system has to fulfill, based on the tuning method proposed by Monje *et al* [9]. Gain margin (GM) and phase margin (PM) are located in the first subsection **Gain and phase margins**, followed by the two functions of sensitivity to noise and disturbances on **Noise and disturbance rejection**. The third division, **Critical frequency and gain variation robustness** is used to enable the robustness against variations in the gain of the plant, hence the crossover frequency ω_c is specified there, altogether determines the bandwidth of the system. The subsections are activated independently, in our case all three are used to fully exploit the optimization process. The performance specifications, taken from [1] in order to make a comparison, were $GM = 20dB$, $PM = 51.83°$ if $\zeta = 0.5$ (or $MP \simeq 16.3\%$), $T(s) \leq -10dB\ \forall\omega \geq \omega t = 10rad/s$, $S(s) \leq -20dB\ \forall\omega \leq \omega s = 0.01rad/s$



```
>> g2
Fractional-order transfer function:
 1
--- exp(-0.1s)
s+1
𝑓x >>
```

Edit FO ...

System name: g2

Zero polynomial (e.g. s+1): 1

Pole polynomial (e.g. -s^1.5-1): s+1

Delay [sec]: 0.1

FOTF Viewer

Fractional order transfer functions: g2

Export system as... Oustaloup filter zpk

☐ Automatically launch LTI Viewer

Time domain: Step

☐ Show progress bar ☐ Superimpose

t (time) vector: 0:0.1:30

u(t) input vector:

Save simulation result: y

Hint: leave empty to discard the simulation result

Frequency domain: Bode

Frequency range [rad/s]: ☐ Superimpose

Figure 4.4: *Window of the command fomcon and its tools.*

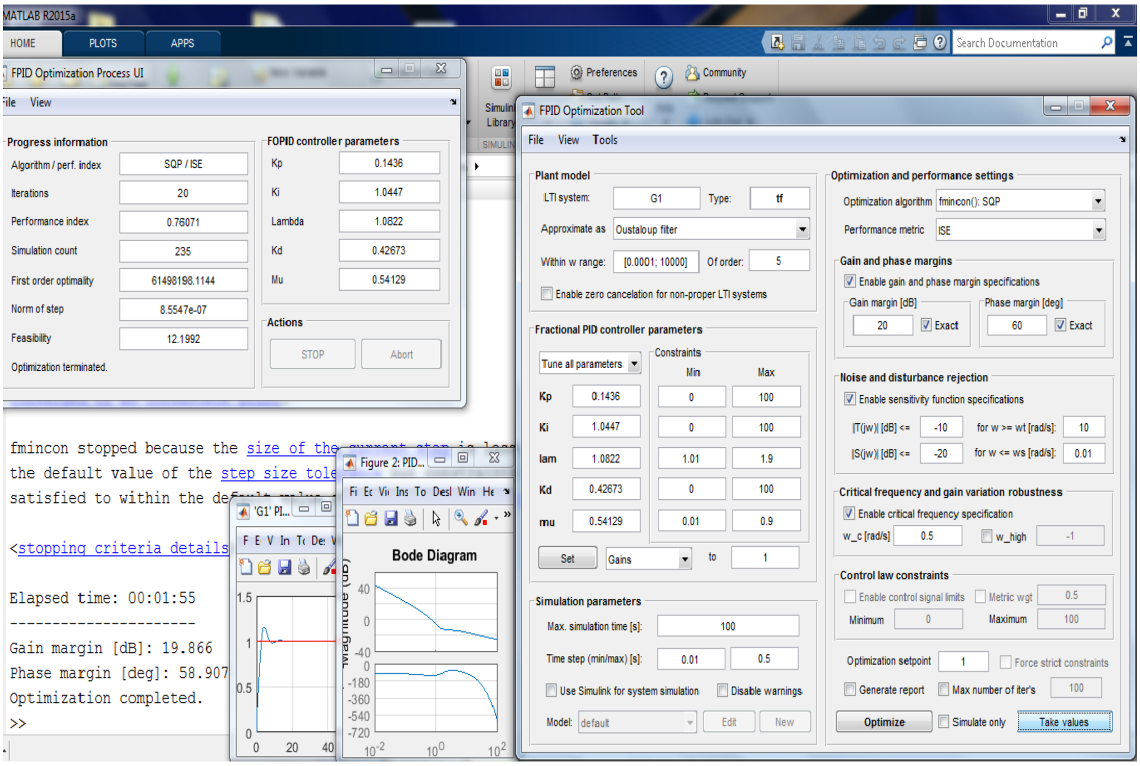
and $\omega_{cg} = 0.5rad/s$.

Figure 4.4 shows another way to declare a transfer function when *fomcon* at the command line is typed if were necessary to work with a fractional type plant or make a model of this kind. The command window shows the same plant but identified as a fractional one since it has been created in the workspace using the tools of the application. The options displayed in this window are very useful to visualize the response of the plant declared in *FOTF Viewer*, both in time and frequency.

4.3.4 Verification of the tuning

At the end of the optimization process, once the "Optimize" button of Figure 4.3 is clicked with the conditions described in section 4.3.3, the program indicates on the command window the margins of phase and gain reached and how long takes to conclude. Values are updated as shown in Figure 4.5 and when selecting the option ”Take values” on Figure 4.3 the optimization interface saves the controller parameters and opens a new window that can simulate the response in time and frequency of the plant-controller. It is shown in Figure 4.6 that this window has other tools, making each FOMCON’s interface a dynamic tool for the design and simulation of fractional systems, from such data we can simulate the final behavior of the system [38]. Figures 4.7 and 4.8 respectively illustrates if the tuning goal has been reached, in the case of the transient response it has greater overshoot ($Mp = 28\%$) than desired (16%); the frequency response complies with the specification $MG = 20dB$ and $MP = 51.83°$.

Generally speaking the criteria are met, for finer results one could try different initial values or modify the valid range or, even, predefine certain gains and powers.



FPID Optimization Process UI

Progress information

Algorithm / perf. index	SQP / ISE
Iterations	20
Performance index	0.76071
Simulation count	235
First order optimality	61498198.1144
Norm of step	8.5547e-07
Feasibility	12.1992

Optimization terminated.

FOPID controller parameters

Kp	0.1436
Ki	1.0447
Lambda	1.0822
Kd	0.42673
Mu	0.54129

```
fmincon stopped because the size of the
the default value of the step size tole
satisfied to within the de

<stopping criteria details

Elapsed time: 00:01:55
----------------------
Gain margin [dB]: 19.866
Phase margin [deg]: 58.907
Optimization completed.
>>
```

FPID Optimization Tool

Plant model

LTI system: G1 Type: tf

Approximate as: Oustaloup filter

Within w range: [0.0001; 10000] Of order: 5

☐ Enable zero cancelation for non-proper LTI systems

Fractional PID controller parameters

Tune all parameters

		Constraints Min	Max
Kp	0.1436	0	100
Ki	1.0447	0	100
lam	1.0822	1.01	1.9
Kd	0.42673	0	100
mu	0.54129	0.01	0.9

Set Gains to 1

Simulation parameters

Max. simulation time [s]: 100

Time step (min/max) [s]: 0.01 0.5

☐ Use Simulink for system simulation ☐ Disable warnings

Model: default

Optimization and performance settings

Optimization algorithm: fmincon(): SQP

Performance metric: ISE

Gain and phase margins

☑ Enable gain and phase margin specifications

Gain margin [dB]: 20 ☑ Exact

Phase margin [deg]: 60 ☑ Exact

Noise and disturbance rejection

☑ Enable sensitivity function specifications

|T(jw)| [dB] <= -10 for w >= wt [rad/s]: 10

|S(jw)| [dB] <= -20 for w <= ws [rad/s]: 0.01

Critical frequency and gain variation robustness

☑ Enable critical frequency specification

w_c [rad/s] 0.5 ☐ w_high -1

Control law constraints

☐ Enable control signal limits ☐ Metric wgt 0.5

Minimum 0 Maximum 100

Optimization setpoint 1 ☐ Force strict constraints

☐ Generate report ☐ Max number of iter's 100

☐ Simulate only

Figure 4.5: *Window with the optimization results and simulation data.*

4.4 Neural networks tuning

Estimating a function using a restricted set of samples or input-output data pairs without explicit knowledge of the shape of the function is a required skill in many situations, with the consequent application of efficient techniques in function approximation, pattern recognition, artificial intelligence, machine learning and data mining [**?**, 49].

One such technique is known as artificial neural networks which aims to achieve a model or generalization network by training through a collection of input and output data as a observation of independent events.

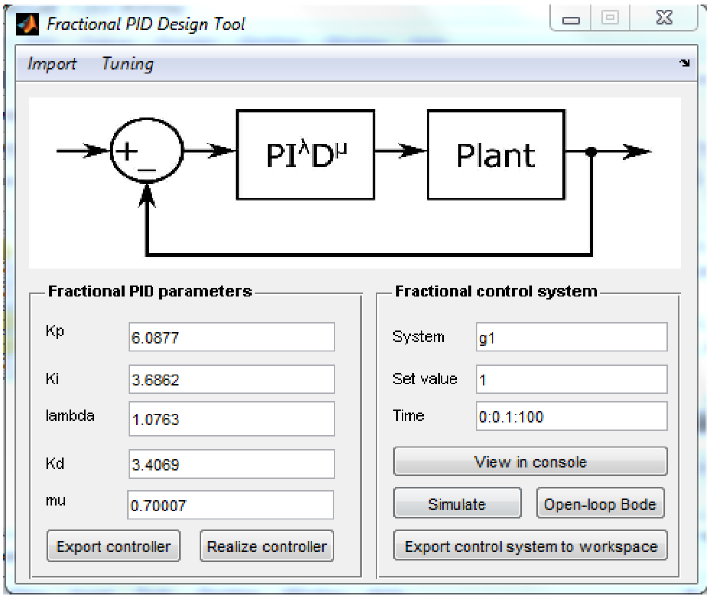
Consider a fractional order PID controller and a plant of the form

$$G(s) = \frac{1}{Ts+1}e^{-Ls} \tag{4.6}$$

The independent variables T and L are the two inputs of five neural networks, one for each parameter K_p, K_i, λ, K_d and μ of the fractional order PID controller. This way, by establishing a model that approximates the value of each controller parameter from any pair (T, L) it is possible to allow an untrained user tune the controller for a particular type of plant. That is the intention of using neural networks, a structure formed by a set of algebraic equations, where the processing functions or neurons are radial basis functions.

The proposed design procedure is as follows:

1. Generate an ordered set of input $[T, L]$ and output data $[K_p, K_i, \lambda, K_d, \mu]$, where the output data is obtained using the five parameter tuning method discussed in section



Fractional PID Design Tool
Import
Tuning
+
_
PI^λD^μ
Plant
Fractional PID parameters
Fractional control system
View in console
Simulate
Open-loop Bode
Export controller
Realize controller
Export control system to workspace

Kp 6.0877
Ki 3.6862
lambda 1.0763
Kd 3.4069
mu 0.70007

System g1
Set value 1
Time 0:0.1:100

Figure 4.6: *Design Tool window with final system parameters.*

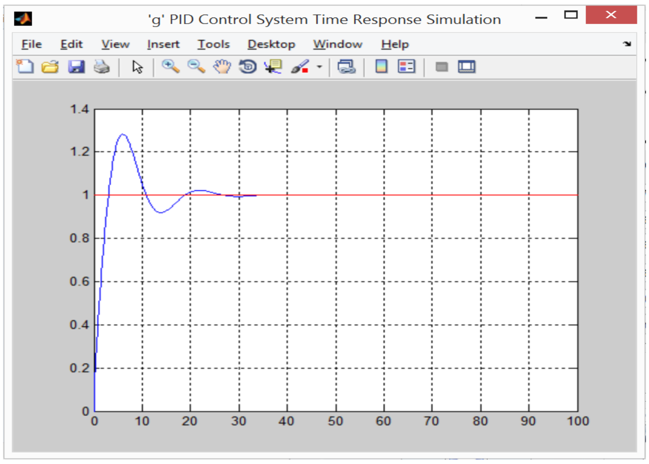


Figure 4.7: *Time response of the fractional system, Mp* = 28%.

2.5.1.

2. Define the minimum and maximum number of neurons that are employed in the neural network.

3. Define the characteristics of the Gaussian function: width, center position and slope factor.

4. Choose the optimization algorithm.

5. Train the network to determine the interconnections among neurons: weights and biases.

6. Validate the model by measuring the approximation error and using validation data.

7. Draw an algebraic network representation if necessary.

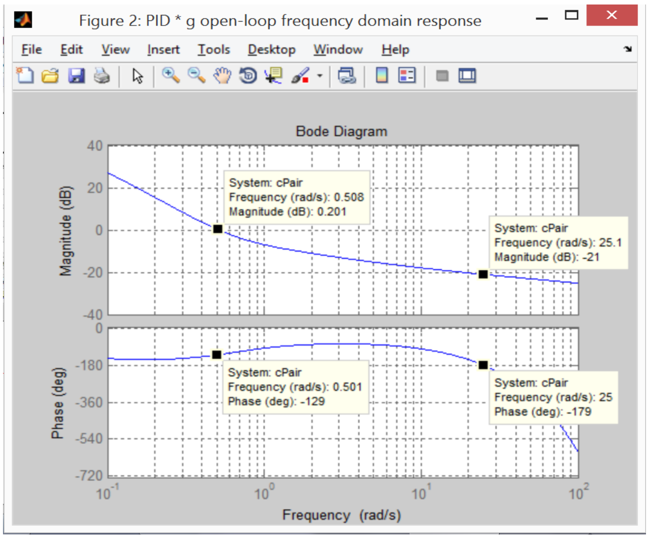


Figure 4.8: *Bode diagram, MG = 21dB and MP = 51°*

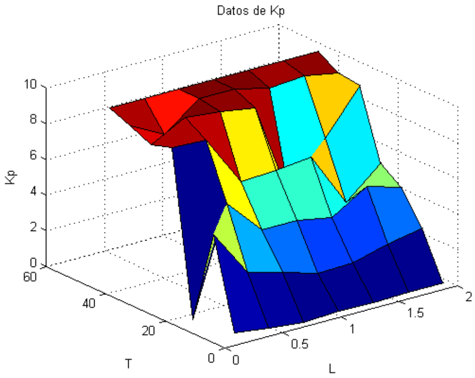


Figure 4.9: *Three-dimensional relation of $K_p, K_i, \lambda, K_d, \mu$ with pairs (T_i, L_j)*

4.4.1 Getting data

Consider the plant 3.13 and the set of restrictions described in section 2.5.1 or 4.3. In order to establish the set of ordered input-output data the fractional controller is tuned. Arrays of size 2xN and 5xN are generated, corresponding to the plant features and the fractional controller, where N is the number of observations made. In this work N=49 is considered.

Using the same plant model the workspace has been divided as follows: $N1 = 7 \times 7 = 49$ cases; $T1 = \{1, 8, 15, 22, 29, 36, 43\}$; $L1 = \{0.1, 0.4, 0.7, 1.0, 1.3, 1.6, 1.9\}$

Table 4.1 contains the relation between pairs of T and L with K_p, K_i, λ, K_d and μ. Figure **??** group those parameters rendering continuous surfaces in a three-dimension space.

4.4.2 Neural network design in MATLAB

On the command window it is typed *nntool* to display an interface. To start using neural networks in MATLAB data is imported in a vector form then the kind of network

Table 4.1: *Relation of the five FOPID controller parameters with pairs* (T_i, L_j)

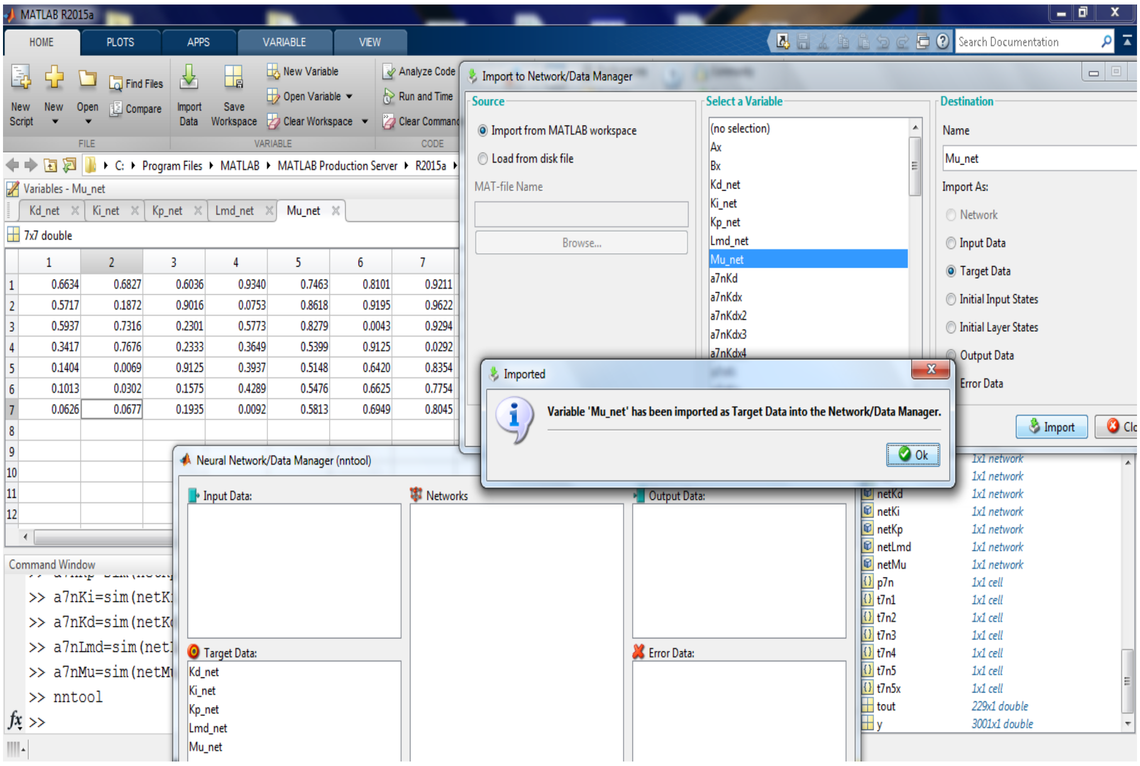
Kp	0.1	0.4	0.7	1.0	1.3	1.6	1.9
1	0.60333	0.3484	0.154	0.23912	0.20328	0.26002	0.23801
8	5.6152	2.9655	2.3975	1.8578	1.8903	2.4408	2.7885
15	0.30247	6.2516	4.2691	4.2656	3.8724	5.1338	4.5442
22	9.4025	9.294	6.4113	6.518	6.7452	3.6714	5.39
29	9.0355	10	9.8105	9.4755	3.8212	6.753	9.1943
36	9.5196	8.5074	10	10	9.5616	9.7211	9.3159
43	10	10	10	10	10	10	10

Ki	0.1	0.4	0.7	1.0	1.3	1.6	1.9
1	0.52411	0.53382	0.57744	0.50905	0.40359	0.38472	0.33274
8	0.81669	1.2436	0.86776	0.52087	0.52101	0.5405	0.58286
15	2.5761	2.3174	1.0188	0.95277	0.78609	0.54355	0.6131
22	2.9562	3.1164	1.3416	1.2586	0.88204	0.51597	0.30284
29	4.3938	3.0152	2.5653	1.7286	1.138	0.82363	0.9305
36	5.2305	3.6232	2.0191	1.5606	1.3252	1.2844	1.0509
43	4.93	4.223	2.4086	1.7781	1.3547	1.0232	1.0476

λ	0.1	0.4	0.7	1.0	1.3	1.6	1.9
1	0.0015889	0.090862	0.1372	0.13801	0.15178	0.22193	0.25044
8	0.01	0.01	0.52378	0.01	6.8313	1.046	1.8135
15	9.9688	0.77973	0.01	1.0465	1.1398	0.01	2.3951
22	0.026852	1.0069	0.01	1.4039	0.035807	1.035	3.7588
29	2.723	0.91653	3.4366	1.9071	6.4571	1.7578	3.7982
36	6.7523	5.0476	0.8284	0.70345	1.9071	4.5959	4.816
43	9.3444	6.1981	3.2334	2.6372	2.6143	2.0845	4.3079

Kd	0.1	0.4	0.7	1.0	1.3	1.6	1.9
1	1.079	1.0149	1.01	1.0118	1.01	1.01	1.0148
8	1.01	1.01	1.01	1.01	1.01	1.0317	1.0407
15	1.01	1.01	1.01	1.01	1.01	1.01	1.01
22	1.01	1.01	1.01	1.01	1.01	1.01	1.01
29	1.01	1.01	1.01	1.01	1.01	1.01	1.01
36	1.01	1.01	1.01	1.01	1.01	1.01	1.01
43	1.01	1.01	1.01	1.01	1.01	1.01	1.01

μ	0.1	0.4	0.7	1.0	1.3	1.6	1.9
1	0.66343	0.73174	0.6223	0.89977	0.89479	0.87995	0.83193
8	0.60856	0.01	0.89664	0.01	0.88383	0.89391	0.9
15	0.28551	0.88336	0.01	0.9	0.74381	0.01	0.88923
22	0.34164	0.9	0.01	0.9	0.043406	0.9	0.077545
29	0.01	0.01	0.9	0.9	0.01	0.9	0.87963
36	0.080977	0.01	0.01	0.01	0.87855	0.9	0.9
43	0.01	0.01	0.01	0.01	0.9	0.01	0.9



C: ▸ Program Files ▸ MATLAB ▸ MATLAB Production Server ▸ R2015a ▸

Variables - Mu_net

Kd_net | Ki_net | Kp_net | Lmd_net | Mu_net

7x7 double

	1	2	3	4	5	6	7
1	0.6634	0.6827	0.6036	0.9340	0.7463	0.8101	0.9211
2	0.5717	0.1872	0.9016	0.0753	0.8618	0.9195	0.9622
3	0.5937	0.7316	0.2301	0.5773	0.8279	0.0043	0.9294
4	0.3417	0.7676	0.2333	0.3649	0.5399	0.9125	0.0292
5	0.1404	0.0069	0.9125	0.3937	0.5148	0.6420	0.8354
6	0.1013	0.0302	0.1575	0.4289	0.5476	0.6625	0.7754
7	0.0626	0.0677	0.1935	0.0092	0.5813	0.6949	0.8045
8							
9							
10							
11							
12							

Import to Network/Data Manager

Source
● Import from MATLAB workspace
○ Load from disk file
MAT-file Name
Browse...

Select a Variable
(no selection)
Ax
Bx
Kd_net
Ki_net
Kp_net
Lmd_net
Mu_net
a7nKd
a7nKdx
a7nKdx2
a7nKdx3
a7nKdx4

Destination
Name: Mu_net
Import As:
○ Network
○ Input Data
● Target Data
○ Initial Input States
○ Initial Layer States
○ Output Data
○ Error Data

Import | Close

Imported

Variable 'Mu_net' has been imported as Target Data into the Network/Data Manager.

Ok

Neural Network/Data Manager (nntool)

Input Data:

Networks

Output Data:

Target Data:
Kd_net
Ki_net
Kp_net
Lmd_net
Mu_net

Error Data:

	1x1 network
	1x1 network
netKd	1x1 network
netKi	1x1 network
netKp	1x1 network
netLmd	1x1 network
netMu	1x1 network
p7n	1x1 cell
t7n1	1x1 cell
t7n2	1x1 cell
t7n3	1x1 cell
t7n4	1x1 cell
t7n5	1x1 cell
t7n5x	1x1 cell
tout	229x1 double
y	3001x1 double

Command Window

```
>> a7nKi=sim(netK
>> a7nKd=sim(netK
>> a7nLmd=sim(net
>> a7nMu=sim(netM
>> nntool
fx >>
```

Figure 4.10: *Data importing window of MATLAB*

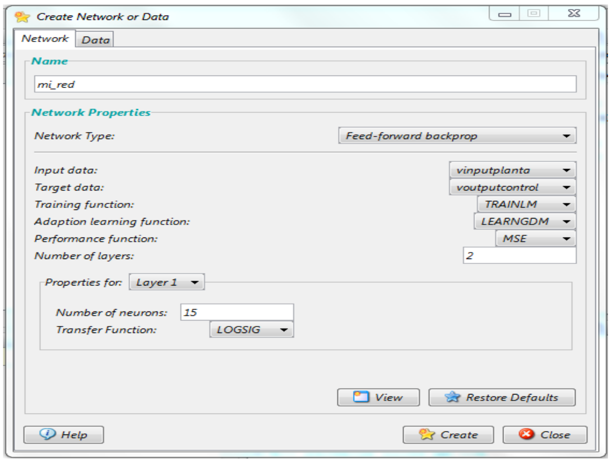
we want to build is chosen. The MATLAB toolbox employs by default a nonlinear function in the hidden layer and one linear in the output layer. The network settings comprises defining the training algorithm, the number of neurons in the hidden layer and the stop criterion [72–74]. Figure 4.10 refers to the selection of both vectors, the input to plant and output controller vector, always is necessary to use this tool, during the importing process as at the end once one have results to save of the trained network.

After the configuration of the network is done, you are asked to run the network to create it in the workspace and start the training. Figure 4.11 shows the appearance of the neural network configuration interface where some modifications could be done, however, this is an option because declaring the network from the *command window* is usual.

During the training one would expect the network identifies the outline of the corresponding function to output vector from the input data. When the number of used neurons as the evaluations increases, the error gradually decreases to a desired objective, ideally null. The image in Figure 4.12 shows the decline in the error value according to the progress in training.

If the results show a poor performance the training can be re-executed where a random portion of the data will be taken mostly to train the network and the rest to make a test of the model accuracy until network parameters found are satisfactory.

When saving the final network parameters in MATLAB's workspace it means that a useful model remains and can be used to obtain fresh original data. Then, new information can be extracted at command line. An array with the pair of T and L desired is sent through the network input so the result is an output vector with five controller



Name

mi_red

Network Properties

Network Type: Feed-forward backprop

Input data: vinputplanta

Target data: voutputcontrol

Training function: TRAINLM

Adaption learning function: LEARNGDM

Performance function: MSE

Number of layers: 2

Properties for: Layer 1

Number of neurons: 15

Transfer Function: LOGSIG

Figure 4.11: *Neural network design in the NN Toolbox*

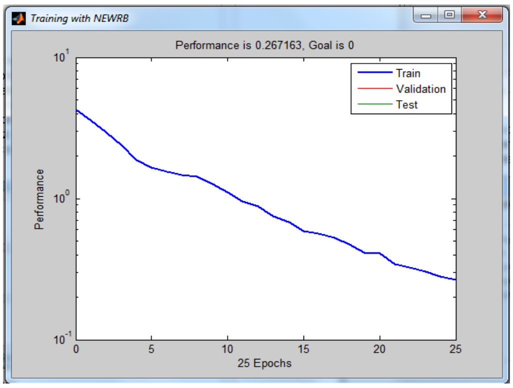


Figure 4.12: *Training result showing the convergence to the goal*

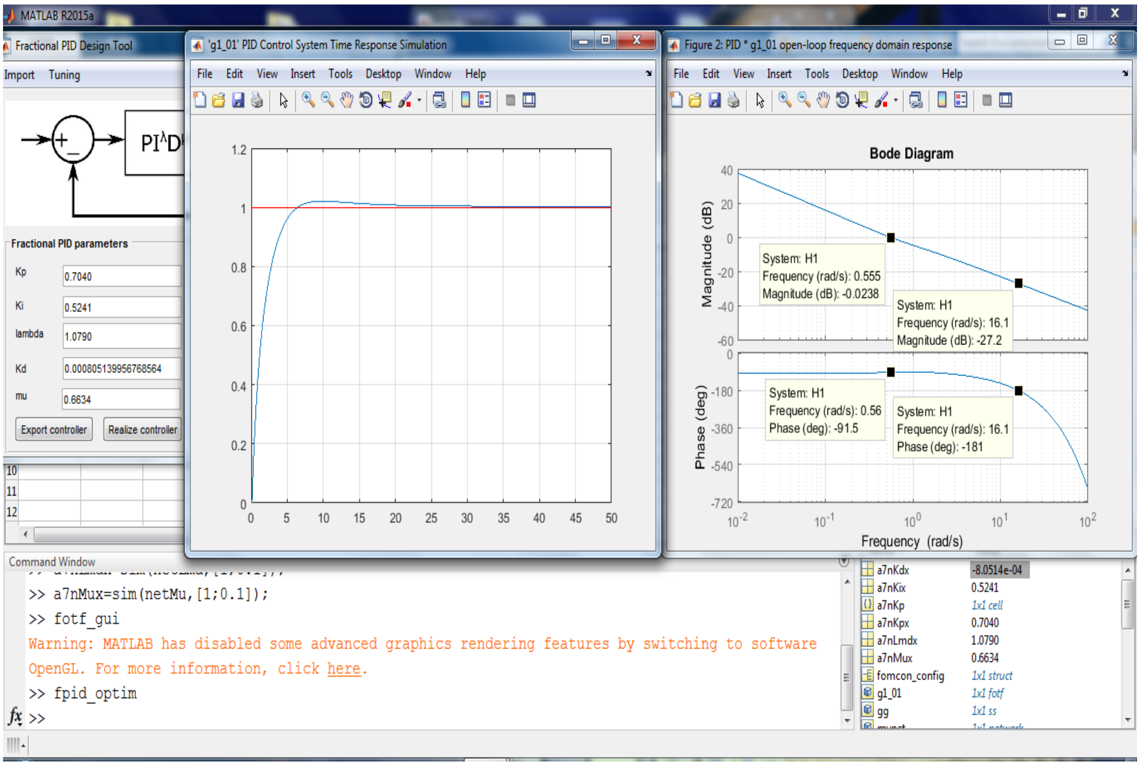
parameters that meet the design specifications.

With the training results saved, the 5 parameters of a FOPID controller can be calculated for any particular case. Here five networks were made and must be imported from the saved file to the *workspace* in order to calculate new outputs from the trained network. Figure 4.13 indicates the view of the window once the characteristics of the network has been established and the training successfully performed. It also shows the validation of the data obtained in terms of setting time and overshoot in the transient response as the frequency margins with the aid of FOMCON.

Figure 4.14 also shows how new data from some specific cases of the tuned FOPID controller parameters are validated. In this work the validation process is performed in at least four ways: using FOMCON tools, using Simulink according to our block approximation, with Spice simulations and discrete circuit test.

In Figure 4.15 are located the responses in time for the simulations in MATLAB Simulink and LTSpice of the system described in [1] and the results obtained with the network tuning for the same system.

On the left side there are four graphs, the first in yellow is the input, the second in purple is the response of the system with an integer PID, the third in blue was obtained in the former paper and the last in red is the response of the neural network. This has almost zero overshoot and a settling time of 8 seconds.



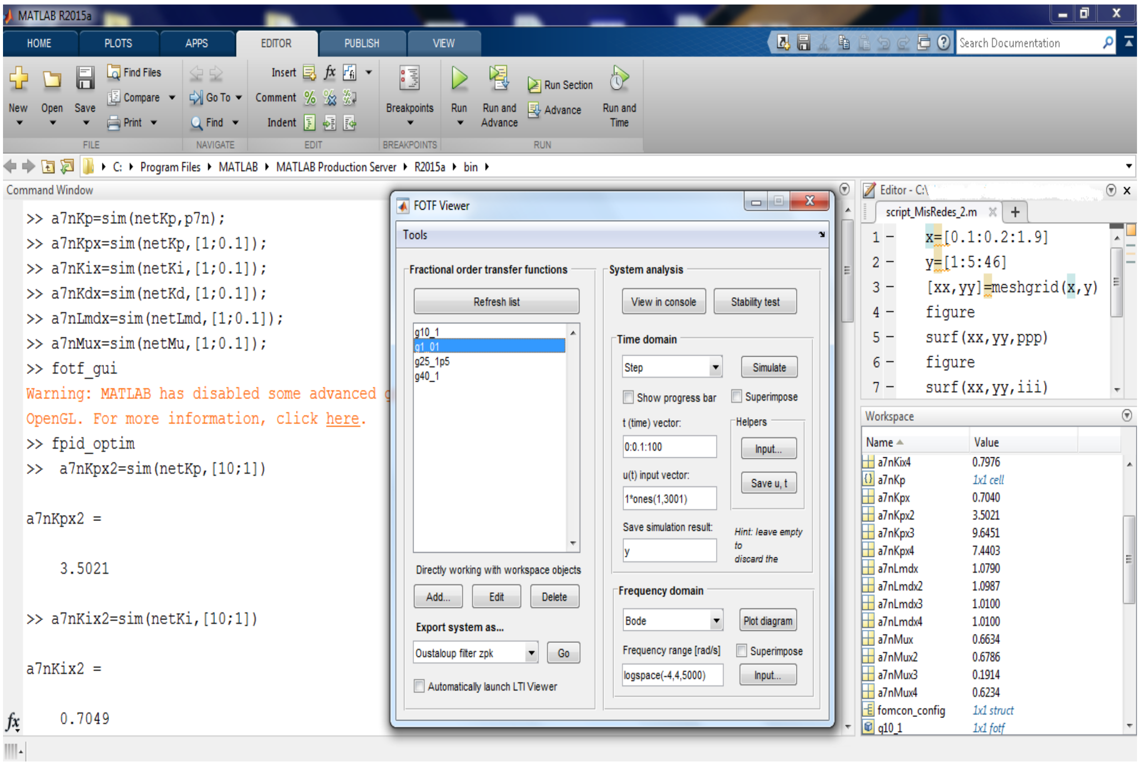
Fractional PID parameters

Kp	0.7040
Ki	0.5241
lambda	1.0790
Kd	0.000805139956768564
mu	0.6634

```
>> a7nMux=sim(netMu,[1;0.1]);
>> fotf_gui
Warning: MATLAB has disabled some advanced graphics rendering features by switching to software
OpenGL. For more information, click here.
>> fpid_optim
>>
```

a7nKdx	-8.0514e-04
a7nKix	0.5241
a7nKp	1x1 cell
a7nKpx	0.7040
a7nLmdx	1.0790
a7nMux	0.6634
fomcon_config	1x1 struct
g1_01	1x1 fotf
gg	1x1 ss

Figure 4.13: *Network configuration window during the training*



```
>> a7nKp=sim(netKp,p7n);
>> a7nKpx=sim(netKp,[1;0.1]);
>> a7nKix=sim(netKi,[1;0.1]);
>> a7nKdx=sim(netKd,[1;0.1]);
>> a7nLmdx=sim(netLmd,[1;0.1]);
>> a7nMux=sim(netMu,[1;0.1]);
>> fotf_gui
Warning: MATLAB has disabled some advanced g
OpenGL. For more information, click here.
>> fpid_optim
>>  a7nKpx2=sim(netKp,[10;1])

a7nKpx2 =

    3.5021

>> a7nKix2=sim(netKi,[10;1])

a7nKix2 =

    0.7049
```

```
x=[0.1:0.2:1.9]
y=[1:5:46]
[xx,yy]=meshgrid(x,y)
figure
surf(xx,yy,ppp)
figure
surf(xx,yy,iii)
```

Name	Value
a7nKix4	0.7976
a7nKp	1x1 cell
a7nKpx	0.7040
a7nKpx2	3.5021
a7nKpx3	9.6451
a7nKpx4	7.4403
a7nLmdx	1.0790
a7nLmdx2	1.0987
a7nLmdx3	1.0100
a7nLmdx4	1.0100
a7nMux	0.6634
a7nMux2	0.6786
a7nMux3	0.1914
a7nMux4	0.6234
fomcon_config	1x1 struct
g10_1	1x1 fotf

Figure 4.14: *Command window output obtained from the neural network model*

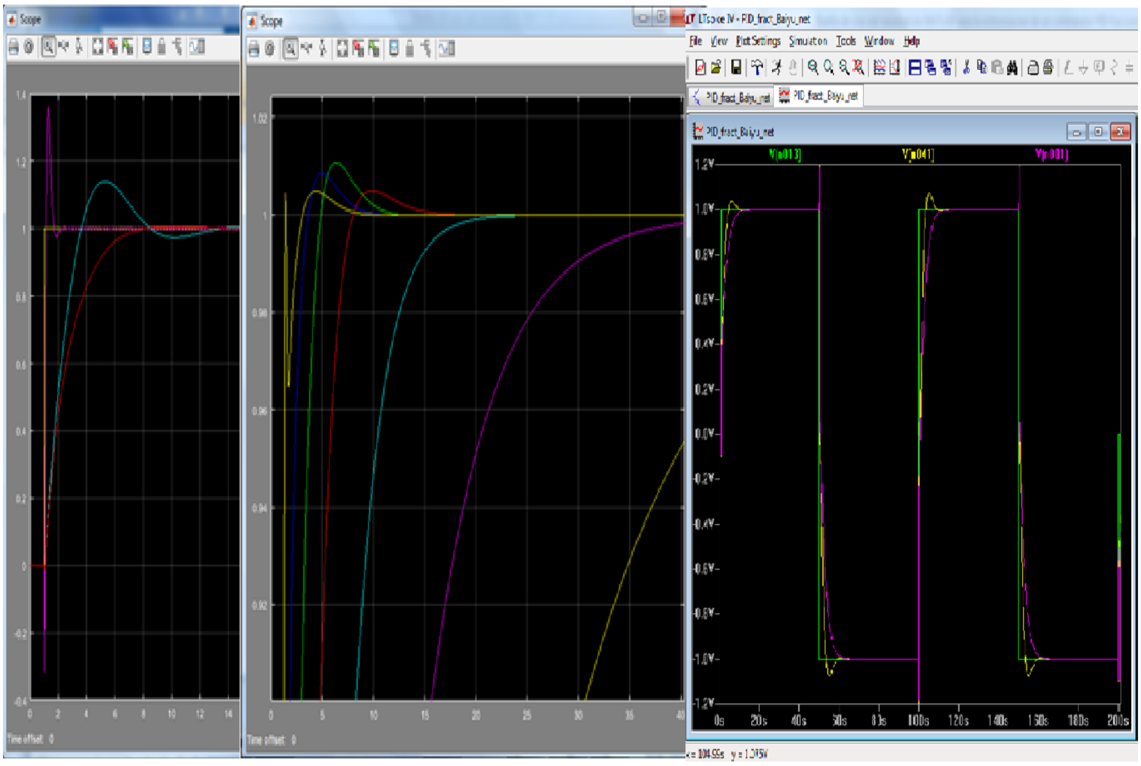


Figure 4.15: *Response in the time obtained with LTSpice of the system tuned to the neural network and the one published by [1]*

In the next box, are the graphs with plant Gain = 8, 4, 2, 1, 0.5, 0.25, 0.125 in the same order of appearance. There is relative uniformity in the overshoot, especially in Gain=8, 1 and Gain=4, 2 of about 1 percent while Gain=0.5, 0.25, 0.125 shows no overshoot neither error for long times.

Finally, on the right side of this image, three graphs are displayed. The green one is the input to the system formed by OpAmps, resistor and capacitors in the corresponding approximation equivalence of the neural networks tuning. The yellow is the response of the system under comparison and the purple line is the response of the proposed tuning. Mp = 7.5 percent and Ts = 8 seconds for the reported system while in our tuning Mp = 0 and Ts = 12 s. With this, we can infer that the simulations in FOMCON, the results applied to block diagrams and the simulations at circuit level are equivalent to the realization with discrete elements and FPAAs.

V

Further applications of fractional Laplace operators

Chapter 5: Further applications of fractional Laplace operators

5.1 Introduction

A methodology for design and implementation of fractional order filters with asymmetric slopes, enhanced selectivity or large notch magnitudes is introduced. This methodology is explained for the case of band-reject response generalized from the second-order to the fractional-order domain and is based on the analog implementation of fractances introduced in Chapter 3, the proposed strategy allows the implementation with commercially- available values of resistances and capacitors.

The most remarkable development in filter theory dates from the early decades of last century [75–77]. As signal processing in electronics engineering became more relevant, this branch of knowledge evolved rendering filter structures capable of accomplish functions whose theoretical background is rigorous and elegant at the same time. Some of the assignments that analog filters can carry on include: frequency duplexing in radar and radio communication systems [78]; impedance matching in power amplifiers [79]; upper-sideband and lower-sideband suppression in upconversion and downconversion mixers [80], respectively; anti-aliasing in data converters [81]; and removal of powerline noise from biomedical signals [82], among others. If difficult trade-offs in their implementation did not limit their usefulness, continuous time filters would be employed in many more applications. Such is the case of analog fractional order filters (FOFs), whose physical realization is rather bulky and hard to accomplish with either commercial component values [83] or in a chip realization with [84]. Probably, the major disadvantage of the reported FOFs is their futility for a practical realization [85–87].

Typically, RC values in the fractor ladder of the FOF are not commercially available. Thus, the implementation of FOFs still is an open problem. Therefore, a design methodology for the implementation of FOFs with asymmetric slopes, enhanced selectivity and large notch magnitudes is introduced in this chapter using the analog implementation of fractances introduced in Chapter 3. Look up tables are provided to perform the entire design of the filter starting from only two specifications: the natural frequency and the quality factor. HSPICE simulation results with Operational Amplifiers (OPAMPs) uA741 of fractional band-reject filter with fractional orders $a = (0.1, 0.3, 0.5, 0.6, 0.9)$ and quality factors $Q \in (1, 650)$ are provided to validate the proposed methodology.

5.2 Band-reject fractional order filter

The fractional-order band-reject filter with asymmetric magnitude characteristic introduced in [88] has the transfer function

$$H_{BR}(s) = \frac{V_{out}(s)}{V_{in}(s)} = \frac{s^2 + r_1}{s^2 + r_2 s^{1+a} + r_3} \tag{5.1}$$

where $0<a<1$ and r_1, r_2, $r_3 \in \mathbb{R}^+$. Contrary to other reported fractional-order band-reject filters (for instance the reported in [89] and [90]), the transfer function (5.1) is

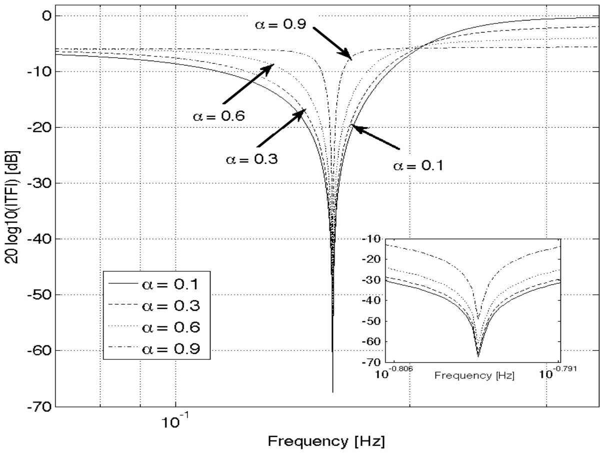


Figure 5.1: *Magnitude response of the band-reject filter for* r_1=r_2=1, r_3=2, *and* a=(0.1, 0.3, 0.6, 0.9).

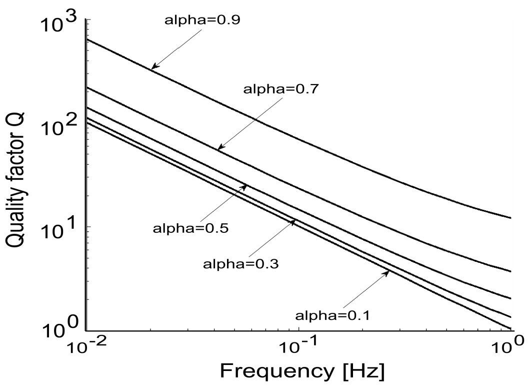


Figure 5.2: *Q factor of the band-reject fractional-order filter versus* r_2 *with* r_1=1, r_3=2 *and* a=(0.1, 0.3, 0.5, 0.7, 0.9).

characterized by simultaneous large values of notch magnitude and quality factor, with a trade-off between these two quantities. The central frequency ω_n and magnitude response $|H_{BR}(j\omega)|$ of (5.1) can be calculated as

$$\omega_n = \sqrt{r_1} \tag{5.2}$$

$$|H_{BR}(\omega)| = \frac{\left|r_1 - \omega^2\right|}{\sqrt{1 + 2\left(\frac{r_3 - \omega^2}{r_2\omega^{1+a}}\right)\sin\left(\frac{a\pi}{2}\right) + \left(\frac{r_3 - \omega^2}{r_2\omega^{1+a}}\right)^2}} \tag{5.3}$$

Figure 5.1 depicts $|H_{BR}(\omega)|$ for the cases r_1=r_2=1, r_3=2, a=(0.1, 0.3, 0.6, 0.9) and $f \in (0.07Hz, 0.4Hz)$. This range includes ω_n normalized to 1rad/s. As can be observed, an asymmetric response with notch magnitudes as large as 60dB is obtained. To calculate numerically the corresponding quality factor Q it is assumed that a can be expressed as a rational number a=$\frac{k}{m}$. Then, it is performed a transformation of the denominator of (5.1), from the s-plane to the W-plane, by means of s=W^m [89]. The resulting characteristic equation becomes

$$W^{2m} + r_2W^{(m+k)} + r_3 = 0 \tag{5.4}$$

A stable system if reached if the argument of all the $2m$ poles of (5.4) satisfy $|\partial_W|>\frac{\pi}{2m}$ [89]. The quality factor is determined from the pair of complex conjugate poles closest to the stability boundary $|\partial_W|=\frac{\pi}{2m}$. These poles, denoted as $\omega_{1,2}$=ω_r±$j\omega_i$, are brought back to the s-plane by means of the transformation $p_{1,2}$=$(\omega_{1,2})^m$. This way, the dominant biquadratic term S_d=$(s-p_1)(s-p_2)$ results

$$S_d = s^2 - 2(\omega_r + \omega_i)^{\frac{m}{2}}\cos\left\{m\arctan\left(\frac{\omega_i}{\omega_r}\right)\right\} + (\omega_r + \omega_i)^m \tag{5.5}$$

which corresponds the quality factor

$$Q = -\frac{1}{2\cos\left\{m\arctan\left(\frac{\omega_i}{\omega_r}\right)\right\}} \tag{5.6}$$

The relationship Q vs r_2, with r_1=1, r_3=2 and a=(0.1, 0.3, 0.5, 0.7, 0.9) is plotted in Fig. 5.2. Values of Q as large as 650 are obtained. It can be observed that Q increases if a increases and r_2 diminishes, i.e., Q increases if the denominator of (5.1) approximates to s^2+r_3, a therm with infinite quality factor.

5.3 Block diagram decomposition

The procedure to approximate the transfer function $H_{BR}(s)$ with integer order integrators is also based on algebraic manipulations of (5.1). The resulting realization, described next, require only three basic building blocks: Weighted Differential Amplifier (WDA), Weighted Two-input Adder Amplifier (WAA) and integer-order Inverter Integrator (IInv).

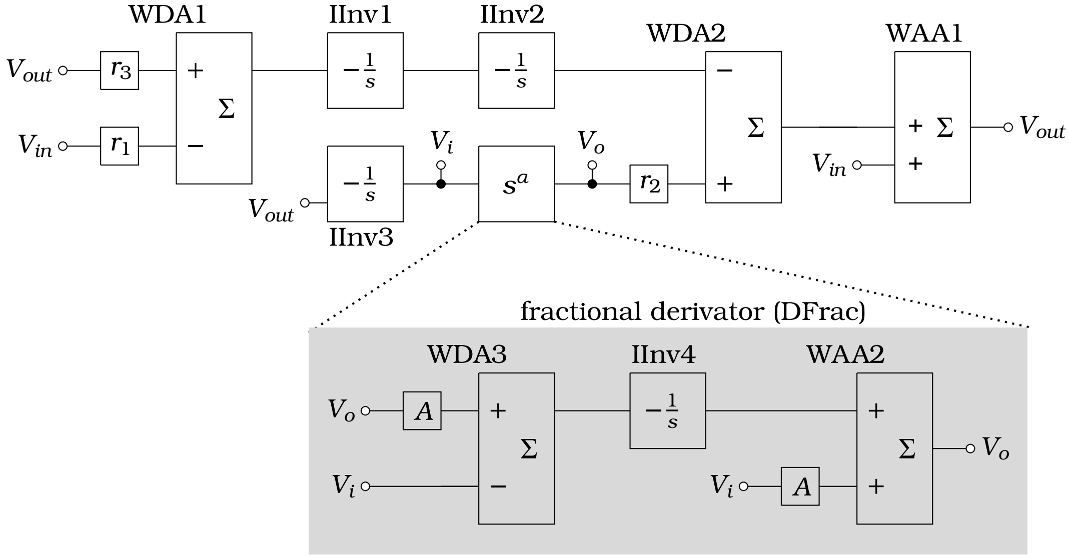


Figure 5.3: *Block diagram of the fractional-order reject-band filter.*

5.3.1 Band-reject filter

Considering (5.1) as the transfer function of a system with input $V_{in}(s)$ and output $V_{out}(s)$ and after performing algebraic manipulation of it is obtained

$$V_{out}(s)\left(s^2 + r_2 s^{1+a} + r_3\right) = V_{in}(s)\left(s^2 + r_1\right) \quad 5.7$$

Now, dividing both sides of (5.7) by s^2 and collecting similar terms results

$$V_{out} = \frac{r_1 V_{in} - r_3 V_{out}}{s^2} - r_2 \frac{s^a}{s} V_{out} + V_{in} \quad 5.8$$

which can be realized with the block diagram representation of Fig. 5.3, composed by two blocks WDA (WDA1 and WDA2), three blocks IInv (IInv1, IInv2 and IInv3), one block WAA (WAA1) and a fractional-order derivator, DFrac, with input V_i and output V_o. The block diagram to implement DFrac is also depicted in Fig. 5.3. It is obtained by considering (3.1) as a transfer function with input V_i and output V_o given by

$$\frac{V_o}{V_i} = \frac{As + 1}{s + A} \quad 5.9$$

Then, by performing algebraic manipulation of (5.9) and after dividing both sides of the resulting equation by s it is obtained

$$V_o = \frac{V_i - AV_o}{s} + AV_i \quad 5.10$$

In agreement with (5.10), the block diagram DFrac can be composed by means of one block WDA, one block IInv and a block WAA (blocks WDA3, IInv4 and WAA2 of Fig. 5.3).

5.4 OpAmp-based building blocks

This section presents OpAmp-based implementations of the block diagrams of Fig. 5.3. This realization are based on the use of the three basic building blocks, WDA, WAA and IInv, which are described next.

5.4.1 Weighted Differential Amplifier (WDA)

It is an electronic amplifier that amplifies the weighted difference between two voltages but does not amplify the particular voltages. Fig. **??** shows a possible implementation with an Operational Amplifier. By nodal analysis of this circuit is calculated the output voltage $V_{out,A}$ as

$$V_{out,A} = \frac{R_g}{R_{g1}} V_{1A} - \frac{R_g}{R_{g2}} V_{2A} \tag{5.11}$$

where R_g can be used as degree of freedom in order to establish the ponderation factors of V_{1A} and V_{2A} by means of resistors R_{g1} and R_{g2}.

5.4.2 Weighted Adder Amplifier (WAA)

This amplifier produces an output voltage $V_{out,B}$ equal to the weighted sum of the two input voltages V_{1B} and V_{2B}. The OpAmp realization of Fig. **??** uses R_g as degree of freedom and R_{h1} and R_{h2} to control the ponderation factors. The output voltage is expressed as

$$V_{out,B} = \frac{R_g}{R_{h1}} V_{1B} + \frac{R_g}{R_{h2}} V_{2B} \tag{5.12}$$

5.4.3 Inverter Integrator (IInv)

The OpAmp based integrator depicted in Fig. **??** uses capacitive feedback to integrate the applied signal. The transfer function of this circuit becomes

$$\frac{V_{out,C}}{V_{1C}} = -\frac{1}{R_x C_x s} = -\frac{1}{s} \tag{5.13}$$

with C_x as degree of freedom and R_X=$1/C_x$.

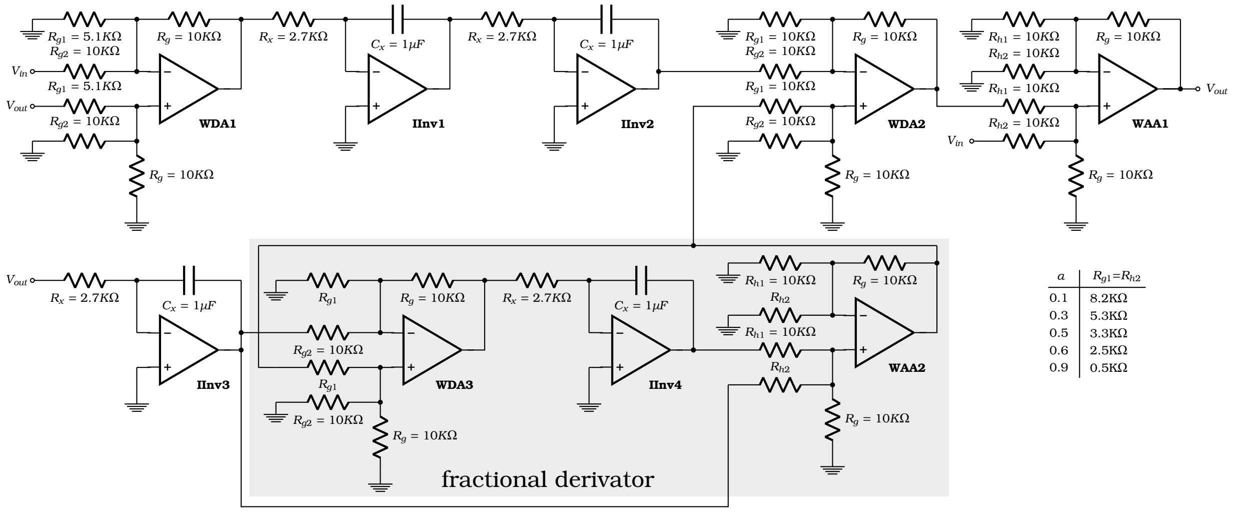
To move ω_n by a factor Ω it can be used the following frequency denormalization in the capacitors of all the employed inverters:

$$C = \frac{C_x}{\Omega} \tag{5.14}$$

5.5 Circuit implementation and design equations

5.5.1 OpAmp-based band-reject filter

The block diagram of Fig. 5.3 can be implemented using the OPAMP-based building blocks of Fig. 7 as is shown in Fig. 5.4. Table 5.1 resume the design equations used to obtain the required values of all the involved resistors, with R_g and C_x as degrees of liberty,



α	R_{g1}=R_{h2}
0.1	8.2KΩ
0.3	5.3KΩ
0.5	3.3KΩ
0.6	2.5KΩ
0.9	0.5KΩ

Figure 5.4: *Discrete implementation of the fractional order reject-band filter.*

Table 5.1: *Design equations for the circuit of Fig. 5.3 with R_g and C_x as degrees of liberty.*

WDA1	WDA2	WDA3	WAA1	WAA2
$R_{g1}=\dfrac{R_g}{r_3}$	$R_{g1}=\dfrac{R_g}{r_2}$	$R_{g1}=\dfrac{R_g}{A}$	$R_{h1}=R_g$	$R_{h1}=R_g$
$R_{g2}=\dfrac{R_g}{r_1}$	$R_{g2}=R_g$	$R_{g2}=R_g$	$R_{h2}=R_g$	$R_{h2}=\dfrac{R_g}{A}$

$A = (1+\alpha)/(1-\alpha)$, $R_x = \frac{1}{C_x}$

Table 5.2: *Design equations for the fractional-order derivator.*

	WDA3 (R_{g2}=R_g)	WAA2 (R_{h1}=R_g)
α=0.1	R_{g1}=0.8181R_g	R_{h2}=0.8181R_g
α=0.2	R_{g1}=0.6666R_g	R_{h2}=0.6666R_g
α=0.3	R_{g1}=0.5384R_g	R_{h2}=0.5384R_g
α=0.4	R_{g1}=0.4285R_g	R_{h2}=0.4285R_g
α=0.5	R_{g1}=0.3333R_g	R_{h2}=0.3333R_g
α=0.6	R_{g1}=0.25R_g	R_{h2}=0.25R_g
α=0.7	R_{g1}=0.1764R_g	R_{h2}=0.1764R_g
α=0.8	R_{g1}=0.1111R_g	R_{h2}=0.1111R_g
α=0.9	R_{g1}=0.0526R_g	R_{h2}=0.0526R_g

and with $R_x = \dfrac{1}{C_x}$ and $A = (1+\alpha)/(1-\alpha)$. These equations were obtained by comparing (5.8) and (5.10) with (5.11), (5.12) and (5.13). Additionally, Table 5.2 presents the design details of the block DFrac (fractional derivator) for different values of the fractional order α. This table was obtained from Table 5.1 by substituting each value of α in $A = (1+\alpha)/(1-\alpha)$.

5.6 Design Methodology

This section formalizes a five-steps methodology for the design and implementation of the band-reject fractional-order filter previously described. This methodology is based on

Fig. 5.1, Fig. 5.2, Table 5.1 and Table 5.2.

- **Step 1**. For the reject-band filter, with r_1=1, r_3=2 and the desired value of Q, obtain r_2 and a from Fig. 5.2.
- **Step 2** The reject-band filter has a normalized frequency ω_n=1rad/s.
- **Step 3** For the reject-band filter, calculate Ω=$\hat{\omega}_n$, where $\hat{\omega}_n$ is the desired (denormalized) center frequency.
- **Step 4**. Select a commercially available value C for all the capacitors of the blocks IInv. Then, calculate C_x=ΩC and chose R_x=$1/C_x$ for all the blocks IInv. It is desired that R_x be on the range of kilo-ohms in order to stablish a good tradeoff between noise contributions and power consumption.
- **Step 5**. To design the blocks WDA, WAA and IInv of Fig. 5.3, chose a commercially available value of R_g and calculate all the resistors R_{g1}, R_{g2}, R_{h1} and R_{h2} with Table 5.1. If a=$(0.1, 0.2, .., 0.9)$ then it can be used the Table 5.2 to design the blocks WDA3 and WAA2.

5.7 Results

OpAmp-based band-reject filter with f_n=60Hz and $Q \geq 2$.

The OpAmp-based realization of the band-reject filter of Fig. 5.4 is composed by 9 OPAMPs, 34 resistors and 4 capacitors. The methodology introduced in Section 5.6 is employed to describe the design of the filter with the following specifications: r_1=1, r_3=2, f_n=60Hz and $Q \geq 2$. Then, the filter is modified to establish a=$(0.3, 0.5, 0.6, 0.9)$.

- **Step 1**. With the desired specifications are obtained, r_2=1 and a=0.5.
- **Step 2**. The normalized natural frequency is ω_n=1rad/s.
- **Step 3**. It is calculated Ω=$2\pi(60Hz)$.
- **Step 4**. The selected commercially available value for the capacitors is C=1μF, resulting C_x=377μF and R_x≈2.7$K\Omega$.
- **Step 5**. For the commercially available resistor R_g=10$K\Omega$ and for each value of a=$(0.3, 0.5, 0.6, 0.9)$ are calculated all the resistors R_{g1}, R_{g2}, R_{h1} and R_{h2} of Fig. 5.4 using the Table 5.1 and the Table 5.2. The design details are summarized in Table 5.3. Note that the values of all capacitors are commercially available and the values of all resistors are commercially available or can be easily approximated with resistor arrays.

The simulation results of the band-reject filter were obtained from simulations performed in HSPICE with the model of the Operational Amplifiers uA741 (rated bandwidth 1MHz and open loop gain of 60dB) and with the design details in Table 5.3. This characteristics shows good agreement with the theoretical response of Fig. 5.5, with deep notch

Table 5.3: *Design details for the circuit of Fig. 5.4.*

	WDA1	WDA2	WDA3	WAA1	WAA2	IInv1, IInv2, IInv3, IInv4
$a = 0.1$	R_{g1}=5$K\Omega$	R_{g1}=10$K\Omega$	R_{g1}=8.2$K\Omega$	R_{h1}=10$K\Omega$	R_{h1}=10$K\Omega$	$R_x = 2.7K\Omega$
	R_{g2}=10$K\Omega$	R_{g2}=10$K\Omega$	R_{g2}=10$K\Omega$	R_{h2}=10$K\Omega$	R_{h2}=8.2$K\Omega$	$C_x = 1\mu F$
$a = 0.3$	R_{g1}=5$K\Omega$	R_{g1}=10$K\Omega$	R_{g1}=5.3$K\Omega$	R_{h1}=10$K\Omega$	R_{h1}=10$K\Omega$	$R_x = 2.7K\Omega$
	R_{g2}=10$K\Omega$	R_{g2}=10$K\Omega$	R_{g2}=10$K\Omega$	R_{h2}=10$K\Omega$	R_{h2}=5.3$K\Omega$	$C_x = 1\mu F$
$a = 0.5$	R_{g1}=5$K\Omega$	R_{g1}=10$K\Omega$	R_{g1}=3.3$K\Omega$	R_{h1}=10$K\Omega$	R_{h1}=10$K\Omega$	$R_x = 2.7K\Omega$
	R_{g2}=10$K\Omega$	R_{g2}=10$K\Omega$	R_{g2}=10$K\Omega$	R_{h2}=10$K\Omega$	R_{h2}=3.3$K\Omega$	$C_x = 1\mu F$
$a = 0.6$	R_{g1}=5$K\Omega$	R_{g1}=10$K\Omega$	R_{g1}=2.5$K\Omega$	R_{h1}=10$K\Omega$	R_{h1}=10$K\Omega$	$R_x = 2.7K\Omega$
	R_{g2}=10$K\Omega$	R_{g2}=10$K\Omega$	R_{g2}=10$K\Omega$	R_{h2}=10$K\Omega$	R_{h2}=2.5$K\Omega$	$C_x = 1\mu F$
$a = 0.9$	R_{g1}=5$K\Omega$	R_{g1}=10$K\Omega$	R_{g1}=0.5$K\Omega$	R_{h1}=10$K\Omega$	R_{h1}=10$K\Omega$	$R_x = 2.7K\Omega$
	R_{g2}=10$K\Omega$	R_{g2}=10$K\Omega$	R_{g2}=10$K\Omega$	R_{h2}=10$K\Omega$	R_{h2}=0.5$K\Omega$	$C_x = 1\mu F$

Table 5.4: *Simulation results of the band-reject filter with f_n= 60Hz, $r_1 = 1$, and $r_3 = 3$ for* a=(0.3, 0.5, 0.6, 0.9)

	f_n	Q	Q
		Simulated	Equation (5.6)
$a = 0.1$	58.67Hz	1.04	2.16
$a = 0.3$	58.68Hz	1.34	2.48
$a = 0.5$	58.68Hz	2.03	3.32
$a = 0.6$	58.68Hz	2.66	4.37
$a = 0.9$	58.68Hz	12.19	14.75

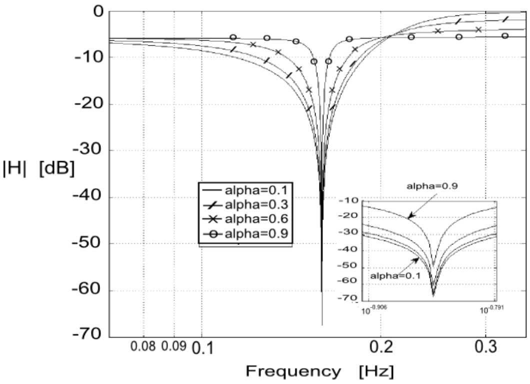


Figure 5.5: *Simulated magnitude response of the reject-band filter for* a=(0.1, 0.3, 0.5, 0.6, 0.9).

magnitudes as large as -56dB. The deviation from the theoretical curve (mainly in the plot corresponding to a=0.9) is attributed to the op-amp non-idealities and the error in the approximation of s^a. Table 5.4 summarizes performance. As can be observed, for the reject-band filter a very similar value to the desired 60Hz was obtained.

The obtained values of Q are above the values obtained by numerical simulation of 5.6, it is attributed to the error introduced by the first-order approximations of s^a. Despite this disadvantage large values of Q were obtained with compact realizations and with commercially available values of resistors and capacitors.

A procedure for the realization of OPAMPs based FOFs and FPAAs based FOFs has been introduced. The usefulness of the suggested approach lies on the use of, solely, commercial value resistors, capacitors and OPAMPs, avoiding the use of inductors and negative impedance converters. The key idea is to manipulate algebraically the network function, including the rational functions who approximate the fractional order integrators and differentiators. The resultant increase of the degrees of freedom allows the rise of the circuit elements with commercial values. In addition, look up tables are provided to perform the entire design of the FOFs of order $(2 - \alpha) \in (0, 2)$ starting from two specifications: the natural frequency f_n , and the quality factor of the filter Q.

VI

Conclusions

An alternative for the analog implementation of integrators and differentiators of fractional order has been introduced. The implementation includes differential amplifiers, adders and integrators of integer order. This proposal was validated simulating (MATLAB and HSPICE) PI and PID controllers of fractional order, and with their realizations using OpAmps and FPAAs. After comparing the results with respect to an equivalent realization from Cauer networks, it was appreciated that in addition to a reduction in the design process, also the number of capacitors and active elements was reduced. Also, the proposal of implementation has the advantage of including degrees of freedom that allow selecting circuit elements or commercial values. These advantages could increase the use of fractional-order PID controllers in industrial applications. Most important is that our proposed implementation for fractional-order integrators and differentiators can be extended to other applications like fractional-order filters, chaotic systems or fractional memristores.

This research also presented a method to carry out OPAMPs based Fractional Order Filters (FOFs) and FPAAs based FOFs. Were attended design examples for the band-pass and band-reject cases. However, the proposed method applies to other network functions, such as low-pass, high-pass and all-pass FOFs. Besides, the suggested method extends the opportunity for a practical realization of Fractional Order Operators used in circuits such as fractional order controllers, fractional order chaotic oscillators and fractional order memristors, to name a few. The usefulness of the proposed approach lies on the use of commercial resistors, capacitors and OPAMPs, avoiding the use of inductors and negative impedance converters. The key idea is to manipulate the network function, including the rational functions that approximate the fractional order integrators and differentiators. The resultant increase of the degrees of freedom allows the rise of the circuit elements with commercial values. In addition, look up tables are provided to perform the full design of the FOFs starting from two specifications: the natural frequency and the quality factor of the filter. Compared to other reported FOFs, the designs synthesized herein required, slightly, more active elements but with passive element values commercially available. The attained Q values are larger than the theoretical expected values, which results from use a first order approach of the fractional order integrators and differentiators. Then, as an alternative, the use of FPAAs was carried out to use their ability of adjusting, on line, the desired parameters of the FOFs.

References

[1] B. O. B. Ou, L. S. L. Song, and C. C. C. Chang, "Tuning of fractional PID controllers by using radial basis function neural networks," *Control and Automation (ICCA), 2010 8th IEEE International Conference on*, pp. 1239–1244, 2010.

[2] C. A. Monje, B. M. Vinagre, A. J. Calderón, V. Feliu, and Y. Q. Chen, "AUTO-TUNING OF FRACTIONAL LEAD-LAG COMPENSATORS," *IFAC Proceedings Volumes*, vol. 38, pp. 319–324, 2005.

[3] R. J. Lian, "Adaptive self-organizing fuzzy sliding-mode radial basis-function neural-network controller for robotic systems," *IEEE Transactions on Industrial Electronics*, 2014.

[4] V. Vesel??, "Robust control methods a systematic survey," 2013.

[5] I. Podlubny, "Fractional-order systems and PI/sup /spl lambda//D/sup /spl mu//-controllers," *IEEE Transactions on Automatic Control*, vol. 44, no. 1, pp. 208–214, 1999.

[6] I. Podlubny, I. Petráš, B. M. Vinagre, P. O'Leary, and L. Dorčák, "Analogue realizations of fractional-order controllers," *Nonlinear Dynamics*, vol. 29, no. 1-4, pp. 281–296, 2002.

[7] B. T. Krishna, "Studies on fractional order differentiators and integrators: A survey," *Signal Processing*, 2011.

[8] A. Tepljakov, E. Petlenkov, and J. Belikov, "Efficient analog implementations of fractional-order controllers," in *Proceedings of the 2013 14th International Carpathian Control Conference, ICCC 2013*, 2013.

[9] C. Monje, B. Vinagre, Y. Chen, V. Feliu, P. Lanusse, and J. Sabatier, "Proposals for Fractional PID Tuning," vol. 024, pp. 2–7, 2005.

[10] C. Z. C. Zhao, D. X. D. Xue, and Y. C. Y. Chen, "A fractional order PID tuning algorithm for a class of fractional order plants," *IEEE International Conference Mechatronics and Automation, 2005*, vol. 1, no. July, pp. 216–221, 2005.

[11] A. Ltifi, M. Ghariani, R. Neji, and A. Integro-differential, "Performance comparison on three parameter determination method of Fractional PID controllers," no. 3, pp. 453–460, 2013.

[12] N. Lachhab, F. Svaricek, F. Wobbe, and H. Rabba, "Fractional Order PID Controller (FOPID)-Toolbox," 2013.

[13] D. V. Dev, "Modified Method of Tuning for Fractional PID controllers," no. January, pp. 8–10, 2014.

[14] P. Shah and S. Agashe, "Review of fractional PID controller," 2016.

[15] Y. Luo, C. Wang, and Y. Chen, "Tuning Fractional Order Proportional Integral Controllers for Fractional Order Systems."

[16] F. Padula and A. Visioli, "Optimal tuning rules for proportional-integer-derivative and fractional-order proportional-integer-derivative controllers for integral and unstable processes," *IET Control Theory & Applications*, 2012.

[17] Z. Yan, J. He, Y. Li, K. Li, and C. Song, "Realization of Fractional Order Controllers by Using Multiple Tuning-Rules," *International Journal of Signal Processing Image Processing and Pattern Recognition*, vol. 6, no. 6, pp. 119–128, 2013. [Online]. Available: http://dx.doi.org/10.14257/ijsip.2013.6.6.12

[18] Y. Q. Chen, I. Petráš, and D. Xue, "Fractional order control - A tutorial," *Proceedings of the American Control Conference*, pp. 1397–1411, 2009.

[19] Monje C., YangQuan Chen, Blas M. Vinagre, Dingyü Xue, and Vicente Feliu, *Fractional-order Systems and Controls*, 2010.

[20] R. Herrmann, *Fractional calculus An Introduction for Physicists*.

[21] T. Tembulkar, S. Darade, S. R. Jadhav, and S. Das, "Design of Fractional Order Differentiator & Integrator Circuit Using RC Cross Ladder Network," *International Journal of Emerging Engineering Research and Technology*, vol. 2, no. 7, 2014.

[22] Antoniou A., "Floating Negative-Impedance Converters," *EEE TRANSACTIONS ON CIRCUIT THEORY*, 1972.

[23] S. Sen and P. K. Dutta, "Realization of a Constant Phase Element and Its Performance Study in a Differentiator Circuit," *IEEE Transactions on Circuits and Systems II: Express Briefs*, 2006.

[24] Pu Yi Fei and Yuan Xiao, "Fracmemristor: Fractional-Order Memristor," *IEEE Access*, 2016.

[25] Jesus Isabel S. and J. A. Tenreiro Machado, "Development of fractional order capacitors based on electrolyte processes," *Nonlinear Dynamics*, 2009.

[26] L. Dorak, J. Terpk, I. Petras, J. Valsa, and E. Gonzalez, "Comparison of the electronic realization of the fractional-order system and its model," in *Proceedings of the 2012 13th International Carpathian Control Conference, ICCC 2012*, 2012.

[27] Astrom K.; Hagglund T., *Control PID avanzado.* PEARSON EDUCACIÓN, 2009.

[28] M. R. Faieghi and A. Nemati, "On Fractional-Order PID Design," *InTechOpen*. [Online]. Available: http://www.intechopen.com/books/applications-of-matlab-in-science-and-engineering

[29] P. Lanusse, J. Sabatier, and A. Oustaloup, "Fractional Order PID and First Generation CRONE Control System Design."

[30] R. Abi Zeid Daou, X. Moreau, and C. Francis, "Study of the effects of structural uncertainties on a fractional system of the first kind - application in vibration isolation with the CRONE suspension," *Signal, Image and Video Processing*, 2012.

[31] C. A. Monje, B. M. Vinagre, Y. Chen, V. Feliu, P. Lanusse, and J. Sabatier, "Optimal Tunings for Fractional P I D Controllers."

[32] J. G. Ziegler and N. B. Nichols, "Optimum settings for automatic controllers," *InTech*, 1995.

[33] Dorf R. and Bishop R., *MODERN CONTROL SYSTEMS*, 2011.

[34] A. Ltifi, M. Ghariani, and R. Neji, "Performance comparison on three parameter determination method of Fractional PID controllers."

[35] D. Valério and J. Sà da Costa, "Introduction to single-input, single-output fractional control," *IET Control Theory & Applications*, 2011.

[36] D. Valério and J. S. da Costa, "Tuning of fractional PID controllers with Ziegler-Nichols-type rules," *Signal Processing*, 2006.

[37] W. Tan, J. Liu, T. Chen, and H. J. Marquez, "Comparison of some well-known PID tuning formulas," *Computers and Chemical Engineering*, vol. 30, pp. 1416–1423, 2006.

[38] A. Tepljakov, E. Petlenkov, and J. Belikov, "A flexible MATLAB tool for optimal fractional-order PID controller design subject to specifications," *Chinese Control Conference, CCC*, no. 3, pp. 4698–4703, 2012. [Online]. Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-84873532217&partnerID=tZOtx3y1

[39] A. T. Azar and F. E. Serrano, "Robust IMC PID tuning for cascade control systems with gain and phase margin specifications," *Neural Computing and Applications*, 2014.

[40] D. Xue, L. Liu, and F. Pan, "Variable-order fuzzy fractional PID controllers for networked control systems," in *Proceedings of the 2015 10th IEEE Conference on Industrial Electronics and Applications, ICIEA 2015*, 2015.

[41] P. Varshney and S. K. Gupta, "Implementation of fractional Fuzzy PID controllers for control of fractional-order systems," in *Proceedings of the 2014 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2014*, 2014.

[42] C. Jáuregui, M. A. Duarte-Mermoud, R. Oróstica, J. C. Travieso-Torres, and O. Beytía, "Conical tank level control using fractional order PID controllers: a simulated and experimental study," *Control Theory and Technology*, 2016.

[43] N. Lachhab, F. Svaricek, F. Wobbe, and H. Rabba, "Fractional Order PID Controller (FOPID) -Toolbox," pp. 3694–3699, 2013.

[44] B. Saidi, M. Amairi, S. Najar, and M. Aoun, “Bode shaping-based design methods of a fractional order PID controller for uncertain systems,” *Nonlinear Dynamics*, 2015.

[45] C. H. Lee and F. K. Chang, “Fractional-order PID controller optimization via improved electromagnetism-like algorithm,” *Expert Systems with Applications*, vol. 37, pp. 8871–8878, 2010.

[46] Z. Wang, Z. Shen, C. Cai, and K. Jia, “Adaptive Control of Wind Turbine Generator System Based on RBF-PID Neural Network.”

[47] Y. Wang, Y. Chenxie, J. Tan, C. Wang, Y. Wang, and Y. Zhang, “Fuzzy Radial Basis Function Neural Network PID Control System for a Quadrotor UAV Based on Particle Swarm Optimization.”

[48] W. Tong-Xu and M. Hong-Yan, “INTRODUCTION The Research of PMSM RBF Neural Network PID Parameters Self-tuning in Elevator.”

[49] CHERKASSKY V. and MULIER F., *LEARNING FROM DATA Concepts, Theory, and Methods*. John Wiley & Sons, 2007.

[50] A. Oustaloup, P. Melchior, P. Lanusse, O. Cois, and F. Dancla, “The CRONE toolbox for Matlab,” in *CACSD. Conference Proceedings. IEEE International Symposium on Computer-Aided Control System Design (Cat. No.00TH8537)*.

[51] R. Matusu, “MATLAB TOOLBOXES FOR FRACTIONAL ORDER CONTROL: AN OVERVIEW,” *Editor B. Katalinic*, vol. 22, no. 1.

[52] D. Valério and J. Sá Da Costa, “NINTEGER: A NON-INTEGER CONTROL TOOLBOX FOR MATLAB,” 2005. [Online]. Available: https://web.ist.utl.pt/duarte.valerio/ninteger/Manual.pdf

[53] A. Tepljakov, E. Petlenkov, and J. Belikov, “FOMCON: Fractional-Order Modeling and Control Toolbox for MATLAB,” 2011.

[54] J. Zhong and L. Li, “Tuning Fractional-Order PI ∂ D μ Controllers for a Solid-Core Magnetic Bearing System,” *IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY*, vol. 23, no. 4, 2015.

[55] S. Khubalkar, A. Chopade, A. Junghare, M. Aware, and S. Das, “Design and Realization of Stand-Alone Digital Fractional Order PID Controller for Buck Converter Fed DC Motor,” *Circuits, Systems, and Signal Processing*, 2016.

[56] S. Mehta and M. Jain, “Comparative analysis of different fractional PID tuning methods for the first order system,” in *2015 1st International Conference on Futuristic Trends in Computational Analysis and Knowledge Management, ABLAZE 2015*, 2015.

[57] Angel L. and Viola J., “Design and Statistical Robustness Analysis of FOPID, IOPID and SIMC PID Controllers Applied to a Motor-Generator System,” *IEEE LATIN AMERICA TRANSACTIONS*, vol. 13, no. 12, pp. 3724–3734, 2015.

[58] Viola J. and Angel L., “Identification, Control And Robustness Analysis Of A Robotic System Using Fractional Control,” *IEEE LATIN AMERICA TRANSACTIONS*, vol. 13, no. 5, pp. 1294–1302, 2015.

[59] D. Valério and J. S. D. Costa, “A review of tuning methods for fractional PIDs,” 2006.

[60] P. Dobra, M. Trusca, and R. Duma, “Embedded application of fractional order control,” *Electronics Letters*, vol. 48, pp. 1526–1528, 2012. [Online]. Available: http://digital-library.theiet.org/content/journals/10.1049/el.2012.1829

[61] Truong Nguyen Luan Vu and Moonyong Lee, “Analytical design of fractional-order proportional-integral controllers for time-delay processes,” *ISA Transactions*, vol. 52, p. 583–591, 2013.

[62] J.-Y. Cao, J. Liang, and B.-G. Cao, “OPTIMIZATION OF FRACTIONAL ORDER PID CONTROLLERS BASED ON GENETIC ALGORITHMS,” pp. 18–21, 2005.

[63] D. Valério and J. Sá Da Costa, “A review of tuning methods for fractional PIDs.”

[64] A. Charef, “Analogue realisation of fractional-order integrator, differentiator and fractional PI.”

[65] “Anadigm Web page.” [Online]. Available: http://www.anadigm.com/an231e04.asp.

[66] G. Tsirimokou, C. Psychalinos, and A. S. Elwakil, “Digitally programmed fractional-order Chebyshev filters realizations using current-mirrors,” in *Proceedings - IEEE International Symposium on Circuits and Systems*, 2015.

[67] C. Sanchez-Lopez, J. Mendoza-Lopez, C. Muniz-Montero, L. A. Sanchez-Gaspariano, and J. M. Munoz-Pacheco, “Accuracy vs simulation speed trade-off enhancements in the generation of chaotic attractors,” in *2013 IEEE 4th Latin American Symposium on Circuits and Systems, LASCAS 2013 - Conference Proceedings*, 2013.

[68] M. E. Fouda and A. G. Radwan, “ON THE FRACTIONAL-ORDER MEMRISTOR MODEL,” *Journal of Fractional Calculus and Applications*, vol. 4, no. 1, pp. 1–7, 2013. [Online]. Available: http://www.fcaj.webs.com/

[69] R. Reddy Mettu and Ramesh. Ganta, “Design of Fractional Order Pi λ d μ Controller for Liquid Level Control of a Spherical Tank Modeled As a Fractional Order System,” *Journal of Engineering Research and Applications www.ijera.com ISSN*, vol. 3, no. 6. [Online]. Available: www.ijera.com

[70] “FOMCON Web page,” 2017. [Online]. Available: http://fomcon.net/fomcon-toolbox/overview/

[71] A. Tepljakov, Eduard Petlenkov, and Juri Belikov, “Fractional-order Calculus based Identification and Control of Linear Dynamic Systems,” Ph.D. dissertation, 2011.

[72] H. N. Koivo, “NEURAL NETWORKS: Basics using MATLAB Neural Network Toolbox,” 2008.

[73] P. Potocnik, “Neural Networks: MATLAB examples Neural Networks course (practical examples) © 2012 Primoz Potocnik Neuron output Neural Networks course (practical examples) © 2012 Primoz Potocnik PROBLEM DESCRIPTION: Calculate the output of a simple neuron.”

[74] H. Demuth and M. Beale, “Computation Visualization Programming Neural Network Toolbox For Use with MATLAB User’s Guide.”

[75] Eastmant A., “The Application of Filter Theory to the Design Of Reactance Networks,” *Proceedings of the I.R.E.*, 1944.

[76] BELEVITCHT V., “Recent Developments in Filter Theory,” *IRE TRANSACTIONS ON CIRCUIT THEORY*.

[77] V. Belevitch, “Summary of the History of Circuit Theory,” *Proceedings of the IRE*, 1962.

[78] L. Xu, D. Feng, and X. Wang, “Matched-filter properties of linear-frequency-modulation radar signal reflected from a phase-switched screen.” [Online]. Available: www.ietdl.org

[79] A. Saberkari, S. Ziabakhsh, H. Martinez, and E. Alarc??n, “Active inductor-based tunable impedance matching network for RF power amplifier application,” *Integration, the VLSI Journal*, 2016.

[80] L. A. Sanchez Gaspariano, A. J. Annema, C. Muniz Montero, and A. D??az S??nchez, “CMOS upconversion mixer with filterless carrier feedthrough cancelation and output power tuning,” *AEU - International Journal of Electronics and Communications*, 2014.

[81] M. Rachid, S. Pamarti, and B. Daneshrad, “Filtering by aliasing,” *IEEE Transactions on Signal Processing*, 2013.

[82] F. Shirbani and S. K. Setarehdan, “ECG power line interference removal using combination of FFT and adaptive non-linear noise estimator,” in *2013 21st Iranian Conference on Electrical Engineering, ICEE 2013*, 2013.

[83] A. Adhikary, S. Sen, and K. Biswas, “Practical Realization of Tunable Fractional Order Parallel Resonator and Fractional Order Filters,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2016.

[84] I. Dimeas, I. Petras, and C. Psychalinos, “New analog implementation technique for fractional-order controller: A DC motor control,” 2017.

[85] A. S. Ali, A. G. Radwan, and A. M. Soliman, “Fractional order butterworth filter: Active and passive realizations,” *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 2013.

[86] A. Soltan, A. G. Radwan, and A. M. Soliman, “CCII based fractional filters of different orders,” *Journal of Advanced Research*, 2014.

[87] J. Baranowski, P. Piatek, and K. Niemczyk, “Design and implementation of non-integer band pass filters for EEG processing,” in *2016 39th International Conference on Telecommunications and Signal Processing, TSP 2016*, 2016.

[88] A. Marathe, B. Maundy, and A. Elwakil, “Design of fractional notch filter with asymmetric slopes and large values of notch magnitude,” in *Midwest Symposium on Circuits and Systems*, 2013.

[89] A. G. RADWAN, A. S. ELWAKIL, and A. M. SOLIMAN, “ON THE GENERALIZATION OF SECOND-ORDER FILTERS TO THE FRACTIONAL-ORDER DOMAIN,” *Journal of Circuits, Systems and Computers*, 2009.

[90] T. J. Freeborn, B. Maundy, and A. Elwakil, “Fractional resonance-based RL??C?? filters,” *Mathematical Problems in Engineering*, 2013.