THESIS ON INFORMATICS AND SYSTEM ENGINEERING C104

Fractional-order Modeling and Control of Dynamic Systems

ALEKSEI TEPLJAKOV

TUT
PRESS

TALLINN UNIVERSITY OF TECHNOLOGY
Faculty of Information Technology
Department of Computer Control

Dissertation was accepted for the defence of the degree of Doctor of Philosophy in Informatics and System Engineering on July 17, 2015.

Supervisors: Ph.D., Associate Professor Eduard Petlenkov, Department of Computer Control, Tallinn University of Technology

Ph.D., Associate Professor Juri Belikov, Department of Computer Control, Tallinn University of Technology

Opponents: D.Sc., Professor Igor Podlubny, Faculty of Mining, Ecology, Process Control and Geotechnologies, Technical University of Košice, Košice, Slovak Republic

Ph.D., Małgorzata Wyrwas, Faculty of Computer Science, Bialystok University of Technology, Białystok, Poland

Defence of the thesis: August 21, 2015

Declaration:
Hereby I declare that this doctoral thesis, my original investigation and achievement, submitted for the doctoral degree at Tallinn University of Technology has not been submitted for any academic degree.

/Aleksei Tepljakov/

Copyright: Aleksei Tepljakov, 2015
ISSN 1406-4731
ISBN 978-9949-23-823-1 (publication)
ISBN 978-9949-23-824-8 (PDF)

INFORMAATIKA JA SÜSTEEMITEHNIKA C104

Murrulistel tuletistel põhinev dünaamiliste süsteemide identifitseerimine ja juhtimine

ALEKSEI TEPLJAKOV

TTÜ
KIRJASTUS

Contents

List of Publications

P1. A. Tepljakov, E. Petlenkov, and J. Belikov, "FOMCON: a MATLAB toolbox for fractional-order system identification and control," International Journal of Microelectronics and Computer Science, vol. 2, no. 2, pp. 51–62, 2011.

P2. A. Tepljakov, E. Petlenkov, J. Belikov, and M. Halás, "Design and implementation of fractional-order PID controllers for a fluid tank system," in Proc. of the 2013 American Control Conference (ACC), Washington DC, USA, June 2013, pp. 1780–1785.

P3. A. Tepljakov, E. Petlenkov, and J. Belikov, "Efficient analog implementations of fractional-order controllers," in Proc. of the 14th International Carpathian Control Conference (ICCC), 2013, pp. 377–382.

P4. A. Tepljakov, E. Petlenkov, and J. Belikov, "Gain and order scheduled fractional-order PID control of fluid level in a multi-tank system," in 2014 International Conference on Fractional Differentiation and its Applications, 2014, pp. 1–6.

P5. A. Tepljakov, E. Petlenkov, and J. Belikov, "Fractional-order digital filter approximation method for embedded control applications," International Journal of Microelectronics and Computer Science, vol. 5, no. 2, pp. 54–60, 2014.

P6. A. Tepljakov, E. Petlenkov, J. Belikov, and E. A. Gonzalez, "Design of retuning fractional PID controllers for a closed-loop magnetic levitation control system," in ICARCV 2014: The 13th International Conference on Control, Automation, Robotics & Vision, 2014, pp. 1345–1350.

P7. A. Tepljakov, E. Petlenkov, and J. Belikov, "FOPID controlling tuning for fractional FOPDT plants subject to design specifications in the frequency domain," in Proc. of the 2015 European Control Conference (ECC), 2015, pp. 3507–3512.

Author's Contribution to the Publications

All results in [P1]–[P7] were obtained by the author of the thesis under the supervision of Dr. Eduard Petlenkov and Dr. Juri Belikov.

In [P1] the foundations for further development of identification and control algorithms for fractional systems in the context of the FOMCON project were established. The initial version of the software framework—FOMCON toolbox for MATLAB/Simulink environment—was developed by the author of this thesis based on available code of the FOTF toolbox. Initial verification of the algorithms was carried out by means of software simulations.

The contribution of the author of the thesis in [P2] lies in the application of the developed FOPID controller tuning algorithms to a laboratory model of an industrial plant. The latter was kindly provided by Dr. Miroslav Halás in the context of a multidisciplinary collaboration under the FOMCON project. Furthermore, a digital implementation of a FOPID controller was produced and verified by real-time experiments. The results related to digital implementation of fractional controllers were further extended in [P5], and a working hardware prototype based on these results was assembled by the author of this thesis.

The problem of implementation of fractional controllers in terms of analog electronic circuits was considered in [P3]. The contribution of the author is the development of a unified approach to generation of fractance circuits that may be used for implementing FO controllers and experimental verification thereof. Since linear controllers in industrial applications are usually limited to a narrow operating range of the real-life nonlinear system, the problem of gain and order scheduling for FOPID controllers was considered in [P4]. The contribution of the author is in the application of a two-point method to the problem of liquid level control and a relevant approach to stability analysis. The method was successfully verified on a laboratory model of an industrial multi tank system.

In [P6] an advanced control method was proposed that was partially based on the results of Dr. Emmanuel Gonzalez. The contribution of the author of the thesis is in the development of an algorithm for stabilizing a nonlinear plant. In particular, an open loop unstable plant was considered—a laboratory model of a magnetic levitation system. The approach is validated through a set of real-time control experiments.

The problem of tuning a FOPID controller for a generalized fractional model of an industrial process is considered in [P7]. The contribution of the author is the development of a numerical method for tuning the controller subject to specifications in the frequency domain. The method was successfully verified using a hardware-in-the-loop real-time experimental platform. In addition, this work establishes the foundation for the development of a hardware FOPID controller prototype with automatic tuning capabilities.

List of Abbreviations

EAP	Electroactive polymer
FFOPDT	Fractional first order plus dead time
FIR	Finite impulse response
FO	Fractional order
FOINVM	Fractional-order inversion model
FOLLC	Fractional lead-lag compensator
FOMCON	Fractional-order Modeling and Control
FOPDT	First order plus dead time
FOPID	Fractional-order proportional integral derivative
FPZP	Fractional power zero-pole
GL	Grünwald-Letnikov
IIR	Infinite impulse response
IPMC	Ion polymer-metal composite
LM	Levenberg-Marquardt
LTI	Linear, time-invariant
NM	Nelder-Mead
PID	Proportional integral derivative
SISO	Single input, single output
SOS	Second-order section
TRR	Trust Region Reflective

Chapter 1

Introduction

> The whole is more than the sum of the parts.
>
> *Aristotle*

Fractional-order calculus offers a novel modeling approach for systems with extraordinary dynamical properties by introducing the notion of a derivative of noninteger (fractional) order. During the last 300 years fractional-order calculus has been the subject of moderately active discussion [23]. The related mathematical theory is well established [51, 69, 87, 96, 101] and provides additional modeling possibilities [93].

In terms of applications, fractional calculus found its way into complex mathematical and physical problems [46, 85]. Heat conduction through a semi-infinite solid [12, 36] and infinite lossy transmission lines [143] are particular examples of fractional systems. Specifically, taking fractional calculus dynamics into account may be useful in modeling any system that possesses memory and/or hereditary properties [101]. Moreover, since fractional calculus is a generalization of conventional calculus, it is expected that fractional models will generally provide a more accurate description of the system dynamics than those based on classical differential equations [72]. Thus, fractional calculus may be conveniently used in many industrial and research fields, e.g., in the study of electrical circuits [43], signal processing [138], chemical processes [85], bioengineering [48] and economic processes [58]. Fractional calculus has been found especially useful in system theory and automatic control, where fractional differential equations are used to obtain more accurate models of dynamic systems, develop new control strategies and enhance the characteristics of control loops. Of significant interest in this regard is industrial process control, and, specifically, the application of fractional process models and fractional-order PID controllers [99], including implementation of automatic tuning thereof. The use of fractional models

and controllers is expected to lead to a significant overall improvement of industrial control loop quality thus providing an increase in control system precision, performance, and energy efficiency [23].

The present thesis is devoted to study of fractional-order calculus based modeling and control of dynamic systems with process control applications. In particular, methods for time and frequency domain identification of fractional order models are proposed and discussed. These methods largely form the basis for model based control design, which constitutes the next part of the thesis, where new methods dealing with optimization of fractional controllers as well as stabilization of unstable systems are presented. Implementation of fractional-order systems and controllers is also investigated, as it is especially important in real-time control applications. All the methods discussed are then presented in the context of a fractional-order modeling and control framework developed for the MATLAB/Simulink environment. Finally, the tools developed in this thesis are applied to real-life control problems. Experiments with laboratory models of real industrial objects are conducted, and the obtained results are analyzed.

1.1 State of the Art

The concept of the differentiation operator $\mathscr{D} = \mathrm{d}/\mathrm{d}x$ is a well-known fundamental tool of modern calculus. For a suitable function f the nth derivative is defined as $\mathscr{D}^n f(x) = \mathrm{d}^n f(x)/\mathrm{d}x^n$, where n is a positive integer. However, what happens if this concept is extended to a situation, when the order of differentiation is arbitrary, for example, $n = 1/2$? That was the very same question L'Hôpital addressed to Leibniz in a letter in 1695. Since then the concept of fractional calculus has drawn the attention of many famous mathematicians, including Euler, Laplace, Fourier, Liouville, Riemann, Abel, and Laurent. But it was not until 1884 that the theory of generalized operators reached a satisfactory level of development for the point of departure for the modern mathematician [69].

The first application of fractional calculus was made by Abel, who in 1823 discovered that the solution of the integral equation for the tautochrone problem could be obtained by means of an integral in the form of a derivative of order 1/2. Later in the 19th century applications of fractional calculus were stimulated by the development of certain mathematical tools by, e.g., Boole and Heaviside. In the 20th century, further contributions in terms of theory and applications have been made by, e.g., Weyl and Hardy, Riesz, Oldham and Spanier [72].

During the last decades of the 20th century, an interest emerged towards the application of fractional calculus in system theory and feedback control. Specifically, the French research group CRONE ("Commande Robuste

d'Ordre Non Entier"[1]) led by Oustaloup [91] developed a set of tools for identification and control of fractional dynamic systems. In particular, the CRONE group is notable for developing time domain identification methods for fractional systems. In [88] a method was proposed to estimate the coefficients of a fractional differential equation based on the Grünwald fractional derivative using a least-squares method. In [128] an identification algorithm for fractional dynamic systems was proposed based on the approximation of a fractional integrator by a rational model. State variable filters and instrumental variable methods were extended for fractional-order system identification in [24]. Recently, optimal instrumental variable methods were extended for fractional systems in [136]. The CRONE research group also developed several generations of CRONE controllers, mostly based on frequency domain analysis of the control system [89,91]. This is also where the well established Oustaloup approximation method for fractional operators originated.

In terms of frequency domain identification methods, a method based on continuous order distributions was proposed in [44], and more recently Levy's identification method was extended to fractional-order systems in [129]. In case of frequency domain identification methods for automatic tuning applications results have been reported in [71,72], where a modified relay feedback-based tuning approach was used for identification of the plant frequency domain characteristics and thereby also for the design of a suitable fractional controller. However, no specific method for recovering a fractional process model was provided.

The fractional-order PID (FOPID) controller was introduced by Podlubny in [98, 99]. It was confirmed that FOPID controllers offer superior performance compared to conventional PID controllers in, e.g., [71,135,147]. A review of tuning methods for FOPID controllers is provided in [131]. These methods can be categorized into analytic, rule-based, and numerical, depending on the approach to the tuning problem. General analytic methods are scarce. Some rule-based methods, partially based on conventional tuning rules were proposed in, e.g., [72, 130]. Numerical methods, on the other hand, have been successfully applied to a number of FOPID controller tuning problems [72]. Global optimization based numerical tuning methods were considered in, e.g., [20, 148].

Since first-order plus dead time (FOPDT) models are frequently used in industrial control design applications [4, 5, 6, 83, 149], the fractional FOPDT (FFOPDT) model and corresponding controller design methods are of significant interest, and have been studied in, e.g., [32,42,61,92,93,105]. Automatic tuning of fractional controllers has not been well developed, however, compared to the case of conventional PID control loops [3].

[1]Noninteger order robust control

Due to the complexity of analytic solutions, or even numerical solutions to fractional differential equations—a recursive process which, in theory, requires an infinite amount of memory—approximations based on the so-called short memory principle are often used. A good overview of existing continuous and discrete time approximations of fractional operators is given in [137]. More recently, methods for analog implementation of fractional-order systems and controllers were proposed in [27, 28, 29]. Digital approximations of fractional operators usually rely on, e.g., power series expansion or continued fraction expansion of corresponding generating functions [23]; other methods based on discretization have been proposed as well [60]. Resulting digital realizations are commonly expressed as finite impulse response (FIR) or infinite impulse response (IIR) filters. Based on this, hardware implementation of fractional-order controllers was discussed in, e.g., [95, 97]. It was noted that since most IIR filter implementations rely on floating-point arithmetic care must be taken to ensure computational stability [23].

Several personal computer software packages were developed to facilitate the task of fractional system identification and controller design. Since the MATLAB/Simulink environment [67] has a wide variety of tools related to optimization, identification and control system analysis and design, authors of existing fractional system modeling and control packages regularly use it as a supporting platform. Among notable packages are CRONE toolbox [90], Ninteger toolbox [132], and FOTF toolbox [23, 72, 146]. The latter served as basis for the FOMCON toolbox [112, 113, 120], developed by the author of this work.

1.2 Motivation and Problem Statement

Based on the state of the art and relevant discussion, the following problems may be formulated and constitute the motivation for the work presented in this thesis.

Fractional calculus allows alleviating the limitations of conventional differential equations where only integer operator powers are used. This gives rise to models of dynamic systems that take into account phenomena such as self-similarity and system state history dependence. Contemporary industrial control systems are of considerable complexity [149], therefore such systems are likely to exhibit such phenomena [72]. Moreover, fractional dynamics have been observed in time domain responses of relatively simple systems [121, 126]. Hence, the application of fractional-order identification and control methods to real industrial control problems is expected to have a positive impact on the particular industrial process in terms of improved performance, efficiency and cost reduction.

It is thus of interest to study control processes with respect to fractional

dynamics. If a process exhibits such dynamics, the model based control design procedure could be carried out using the corresponding tools. This gives rise to the problem of fractional model identification which is related to several issues. These include the choice of an efficient simulation method of the identified fractional model (including process models); choosing the parameters of the model to identify and limiting the number thereof to improve the conditioning of the corresponding optimization problem; the choice of a suitable optimization algorithm for estimating the parameters of the model. To ensure practical usability of the obtained model, methods for its validation with respect to experimental data should be implemented.

Once a valid model of a process is established, one may proceed with model based control design. Fractional dynamics are best compensated with fractional controllers [23, 97]. However, the tuning thereof is more involved compared to conventional controllers. Numerical optimization methods are frequently used to tackle this issue. Due to the complexity of fractional models, the optimization problem must be properly set up.

Due to additional tuning flexibility FOPID controllers are typically capable of outperforming their conventional counterparts, since more design specifications may be fulfilled. Thus, developing a general method for FOPID controller tuning is very desirable [72]. Such a method should be flexible enough to overcome issues with simulating fractional or integer-order models of the control plant in the time domain, and at the same time take into consideration design specifications imposed in the frequency domain to maintain robustness of the controller.

A favorable quality of (FO)PID-type controllers is the realization of robustness criteria that guarantee stability and performance of the control loop under reasonable operating conditions. Modern nonlinear control methods based only on time domain evaluation of system dynamics generally lack this quality due to the complexity of developing a unified robustness concept for a wide enough class of nonlinear systems.

To apply the developed FO control algorithms to specific control problems the corresponding controllers have to be implemented. Two types of realization of such controllers can be proposed: digital implementation and analog implementation. Direct realizations based on mathematical definitions of fractional operators have certain limitations for real-time control [146], which is why approximations of fractional operators are frequently used instead. Several issues may be outlined in connection to this. Namely, computational stability of the signal processing algorithm must be ensured in case of digital implementation, and a feasible set of discrete electronic components must be chosen for the analog implementation of fractance circuits.

As stated above, of particular importance is the use of fractional models and controllers in industrial applications. Studies show that a huge portion of industrial control loops—about 90%—are of PI/PID type; moreover, it was

found that about 80% of these existing control loops are poorly tuned [83]. Since FOPID controllers offer more tuning freedom and stabilizing abilities, it is expected that industrial integration of these controllers will result in considerable benefit. Therefore, a further set of specific research goals may be proposed: to study fractional process models and to provide means for implementation of automatic controller tuning; to develop a gain and order scheduling approach for obtaining a set of FOPID controllers working seamlessly across several operating points; to provide means for stabilizing an unstable fractional or integer-order plants; to investigate the possibilities for incorporation of newly designed fractional controllers, offering superior performance, into existing conventional PI/PID control loops thereby reducing process downtime and related costs; to develop a hardware controller prototype based on developed controller design and synthesis methods.

1.3 Author's Contributions

The main contribution of the author of the thesis is the development of model based control design methods for systems described by fractional dynamic models. This contribution comprises three consecutive parts:

- *Modeling*: Development of methods for time- and frequency domain identification of systems by fractional models [113, 117]. The novelty lies in the implementation of the methods which focus on the model structure to improve the performance and accuracy of the identification algorithm. This is achieved through careful selection of model parameters. The accuracy of the obtained model is assessed by means of statistical methods. Closed-loop identification methods are also proposed to tackle industrial control design problems. A particular algorithm is developed for identifying a process model for automatic FOPID controller tuning applications [122, 123].

- *Control*: Based on the identified model, a general, optimization based novel FOPID controller design method is proposed that uses the Nelder-Mead simplex method to minimize a cost function comprising specifications in both time- and frequency domains [112, 113, 114, 126]. The method can handle both linear and nonlinear system models, and is later used to solve gain and order scheduling [121], stabilization [125], and control loop retuning problems [40, 125]. To the best knowledge of the author, this particular approach has not been used in prior art. In addition, the author has developed FFOPDT model analysis methods and FOPID controller design for automatic tuning applications with focus on embedded device implementation [122, 123]. This part of the contribution extends the results in [71, 72] and presents an alternative approach to solving the automatic FOPID controller tuning problem.

- *Software and hardware implementation*: For practical implementation of the contributions given above a set of methods is developed that either enhance existing methods or leverage their favorable qualities to achieve the desired result. Both analog and digital fractional system realization methods are considered. The most important contribution in this context is the development of a FOPID controller prototype that is used for the implementation and verification of the above mentioned results. The closest known solutions are proposed in [95,97], but do not take into account the specific steps of the digital implementation method proposed in the present contribution. All the main results of this work are also included in a computer-aided control system design software solution—FOMCON toolbox for the MATLAB/Simulink environment—contributed by the author of the thesis [120].

Finally, the proposed FO model based control design methods are verified on a variety of real-life hardware laboratory models of industrial plants.

1.4 Thesis Outline

Each chapter begins with a summary of the research problems discussed therein. Relevant examples are provided where appropriate, including those that span across multiple chapters to illustrate developing ideas. Each chapter of the thesis ends with a section containing concluding remarks pertaining to theoretical and practical results reported in the corresponding chapter. Finally, the last chapter comprises general concluding comments, as well as items for prospective research. In what follows, a summary of each chapter is provided.

Chapter 2

In this chapter the reader is introduced to core concepts of fractional calculus used in dynamic system modeling and control. In addition, an overview of optimization methods used in this work is provided with relevant comments.

Chapter 3

This chapter is devoted to identification of dynamic systems by fractional-order models. Both time- and frequency domain identification methods are considered, and more attention is given to the former case, whereas in the latter case special attention is given to identification of a particular type of process model, as it forms a basis for automatic tuning applications. Open-loop and closed-loop identification methods are also discussed; the latter is based on parametric identification of a fractional model.

Chapter 4

The topic of this chapter is fractional-order control. Specifically, methods for FOPID controller design are discussed, including the following: optimization based tuning subject to specifications in both time- and frequency domains; the gain and order scheduling approach; conventional PID control loop retuning; and stabilization of unstable plants using a randomized method with subsequent optimization. A method for control loop analysis and controller design, considering a particular type of fractional process model, is proposed, with focus on automatic tuning applications.

Chapter 5

In this chapter, analog and digital implementation methods for fractional models and controllers are discussed. Particular attention is given to the digital implementation of the FOPID controller, since the proposed method is used in the developed hardware prototype. In addition, a hardware and software platform used for real-time verification of control algorithms is presented in this chapter.

Chapter 6

This chapter is dedicated to the FOMCON ("Fractional-order Modeling and Control") toolbox for the MATLAB/Simulink environment, which encompasses the majority of the results presented in earlier chapters. The toolbox has a modular structure. The chapter is divided into sections such that each section covers the functions of a particular module. Illustrative examples are provided.

Chapter 7

In this chapter, particular applications of fractional control are provided. Real-life industrial system models are studied. Results presented in Chapters 3 through 5 are applied to the following real-life problems: Fluid level control in a coupled [126] and multiple tank systems [121]; Position control of an unstable plant—a sphere levitating due to an electromagnetic field in a magnetic levitation system [125]; Position control of an ion polymer-metal composite (IPMC) actuator.

Moreover, nonlinear models of the systems under consideration (except for the ion-polymer composite actuator) are obtained, verified, and used for control design purposes. All presented results related to the identification, control, and implementation of fractional systems and controllers are achieved by means of FOMCON toolbox for MATLAB/Simulink described in Chapter 6.

Chapter 2

Preliminaries

In the following chapter the reader is introduced to core concepts and algorithms used in the thesis. The chapter begins with an introduction to fractional calculus tools, including fractional operator definitions and properties. Next, tools related to modeling and analysis of dynamic systems are discussed. Then, fractional-order controllers are introduced. Finally, a brief overview of numerical optimization methods used in this work is provided. Particular attention is given to the treatment of bounded and constrained optimization problems.

2.1 Mathematical Basis

Fractional calculus is a generalization of integration and differentiation to non-integer order operator ${}_a\mathscr{D}_t^\alpha$, where a and t denote the limits of the operation and α denotes the fractional order such that

$$ {}_a\mathscr{D}_t^\alpha = \begin{cases} \frac{\mathrm{d}^\alpha}{\mathrm{d}t^\alpha} & \Re(\alpha) > 0, \\ 1 & \Re(\alpha) = 0, \\ \int_a^t (\mathrm{d}\tau)^{-\alpha} & \Re(\alpha) < 0, \end{cases} \tag{2.1} $$

where generally it is assumed that $\alpha \in \mathbb{R}$, but it may also be a complex number [23]. In this work we consider only the former case.

There exist multiple definitions of the fractional operator [72]. We consider the Grünwald-Letnikov definition, which is used throughout this work for the purpose of numerical solutions to fractional-order differential equations.

Definition 2.1 *(Grünwald-Letnikov)*

$$ {}_a\mathscr{D}_t^\alpha f(t) = \lim_{h \to 0} \frac{1}{h^\alpha} \sum_{k=0}^{\left[\frac{t-a}{h}\right]} (-1)^k \binom{\alpha}{k} f(t - kh), \tag{2.2} $$

where $[\cdot]$ *means the integer part,* h *is the step size.*

Fractional-order differentiation has the following properties [72], [101]:

1. If $f(t)$ is an analytic function, then the fractional-order differentiation ${}_0\mathscr{D}_t^{\alpha} f(t)$ is also analytic with respect to t.
2. If $\alpha = n$ and $n \in \mathbb{Z}_+$, then the operator ${}_0\mathscr{D}_t^{\alpha}$ can be understood as the usual operator $\mathrm{d}^n/\mathrm{d}t^n$.
3. Operator of 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 constants, then
$$ {}_0\mathscr{D}_t^{\alpha} \left[af(t) + bg(t) \right] = a\, {}_0\mathscr{D}_t^{\alpha} f(t) + b\, {}_0\mathscr{D}_t^{\alpha} g(t). \tag{2.3}$$
5. For the fractional-order operators with $\Re(\alpha) > 0, \Re(\beta) > 0$, and under reasonable constraints on the function $f(t)$ it holds the additive law of exponents:
$$ {}_0\mathscr{D}_t^{\alpha} \left[{}_0\mathscr{D}_t^{\beta} f(t) \right] = {}_0\mathscr{D}_t^{\beta} \left[{}_0\mathscr{D}_t^{\alpha} f(t) \right] = {}_0\mathscr{D}_t^{\alpha+\beta} f(t) \tag{2.4}$$
6. The fractional-order derivative commutes with integer-order derivative
$$ \frac{\mathrm{d}^n}{\mathrm{d}t^n} \left({}_a\mathscr{D}_t^{\alpha} f(t) \right) = {}_a\mathscr{D}_t^{\alpha} \left(\frac{\mathrm{d}^n f(t)}{\mathrm{d}t^n} \right) = {}_a\mathscr{D}_t^{\alpha+n} f(t), \tag{2.5}$$
under the condition $t = a$ we have $f^{(k)}(a) = 0, (k = 0, 1, 2, ..., n-1)$.

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

$$ F(s) = \mathscr{L} \left[f(t) \right] = \int_0^{\infty} \mathrm{e}^{-st} f(t) \mathrm{d}t \tag{2.6}$$

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

$$ f(t) = \mathscr{L}^{-1} \left[F(s) \right] = \frac{1}{j2\pi} \int_{c-j\infty}^{c+j\infty} \mathrm{e}^{st} F(s) \mathrm{d}s, \tag{2.7}$$

where c is greater than the real part of all the poles of function $F(s)$ [72].

Assuming zero initial conditions, the Laplace transform of (2.2) is defined as follows.

Definition 2.2 *(Laplace transform of the Grünwald-Letnikov fractional operator)*

$$ \mathscr{L} \left[\mathscr{D}^{\alpha} f(t) \right] = s^{\alpha} F(s). \tag{2.8}$$

2.2 Fractional-order Models

A fractional-order continuous-time dynamic system can be expressed by a fractional differential equation of the following form [72]:

$$\begin{aligned} a_n \mathscr{D}^{\alpha_n} y(t) + a_{n-1} \mathscr{D}^{\alpha_{n-1}} y(t) + \cdots + a_0 \mathscr{D}^{\alpha_0} y(t) \quad &= \\ b_m \mathscr{D}^{\beta_m} u(t) + b_{m-1} \mathscr{D}^{\beta_{m-1}} u(t) + \cdots + b_0 \mathscr{D}^{\beta_0} u(t), & \end{aligned} \tag{2.9}$$

where $(a_i, b_j) \in \mathbb{R}^2$ and $(\alpha_i, \beta_j) \in \mathbb{R}_+^2$. The system is said to be of *commensurate-order* if in (2.9) all the orders of derivation are integer multiples of a base order q such that $\alpha_k, \beta_k = kq, q \in \mathbb{R}^+$. The system can then be expressed as

$$\sum_{k=0}^{n} a_k \mathscr{D}^{kq} y(t) = \sum_{k=0}^{m} b_k \mathscr{D}^{kq} u(t). \tag{2.10}$$

If in (2.10) the order is $q = 1/r, r \in \mathbb{Z}_+$, the system will be of rational order. The diagram with linear time-invariant (LTI) system classification is given in Figure 2.1.

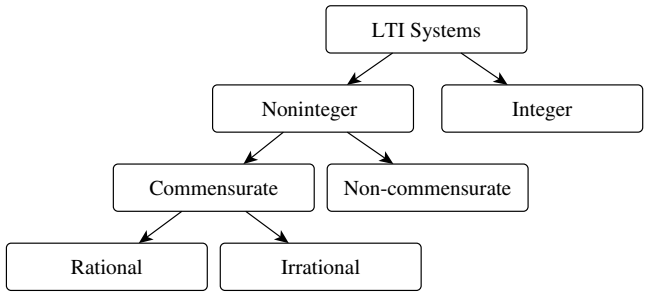


Figure 2.1: Classification of LTI systems

Applying the Laplace transform to (2.9) with zero initial conditions the input-output representation of the fractional-order system can be obtained in the form of a transfer function.

$$G(s) = \frac{Y(s)}{U(s)} = \frac{b_m s^{\beta_m} + b_{m-1} s^{\beta_{m-1}} + \cdots + b_0 s^{\beta_0}}{a_n s^{\alpha_n} + a_{n-1} s^{\alpha_{n-1}} + \cdots + a_0 s^{\alpha_0}}. \tag{2.11}$$

We shall call the number of fractional poles in (2.11) the *pseudo-order* of the system. In the case of a system with commensurate order q, we may take $\sigma = s^q$ and consider the continuous-time pseudo-rational transfer function

$$H(\lambda) = \frac{\sum_{k=0}^{m} b_k \sigma^k}{\sum_{k=0}^{n} a_k \sigma^k}. \tag{2.12}$$

2.2.1 Process Models

In the context of this work we deal with problems of process control. The fractional-order transfer function representation of a process model consists of (2.11) and an input delay term given in the time domain as $u(t) = u_d(t-L)$. The general form is thus

$$G(s) = \frac{b_m s^{\beta_m} + b_{m-1} s^{\beta_{m-1}} + \cdots + b_0 s^{\beta_0}}{a_n s^{\alpha_n} + a_{n-1} s^{\alpha_{n-1}} + \cdots + a_0 s^{\alpha_0}} \mathrm{e}^{-Ls}, \tag{2.13}$$

where it is usual to take $\beta_0 = \alpha_0 = 0$ so that the static gain of the system is given by $K = b_0/a_0$,and $L \in \mathbb{R}_+$.

One particular model of this type is the fractional-order first-order plus delay time model [42, 61, 92]:

$$G_{dt}(s) = \frac{K}{1 + Ts^{\alpha}} \mathrm{e}^{-Ls}, \tag{2.14}$$

where K is the static gain, $L \geqslant 0$ is the delay, $T > 0$ is the time constant, and $\alpha \in (0, 2)$ is the fractional power of the operator. Its conventional counterpart with $\alpha = 1$ has become the basis for numerous tuning rules [83]. Since the generalized version of the process model allows to capture the dynamics of the process under study more accurately, it is expected to be useful in the design of fractional-order controllers [93].

2.2.2 Stability Analysis

In order to determine stability of a fractional system given by (2.10) we consider the following theorem [23, 66].

Theorem 2.1 *(Matignon's stability theorem) The fractional transfer function $G(s) = Z(s)/P(s)$ is stable if and only if the following condition is satisfied in σ-plane:*

$$|\arg(\sigma)| > q\frac{\pi}{2}, \forall \sigma \in C,\ P(\sigma) = 0, \tag{2.15}$$

where $0 < q < 2$ and $\sigma := s^q$. When $\sigma = 0$ is a single root of $P(s)$, the system cannot be stable. For $q = 1$, this is the classical theorem of pole location in the complex plane: no pole is in the closed right plane of the first Riemann sheet.

The algorithm for checking the stability of the system in (2.11) can be summarized as follows:

1. Find the commensurate order q of $P(s)$, find $a_1, a_2, \ldots a_n$ in (2.12);
2. Solve for σ the equation $\sum_{k=0}^{n} a_k \sigma^k = 0$.

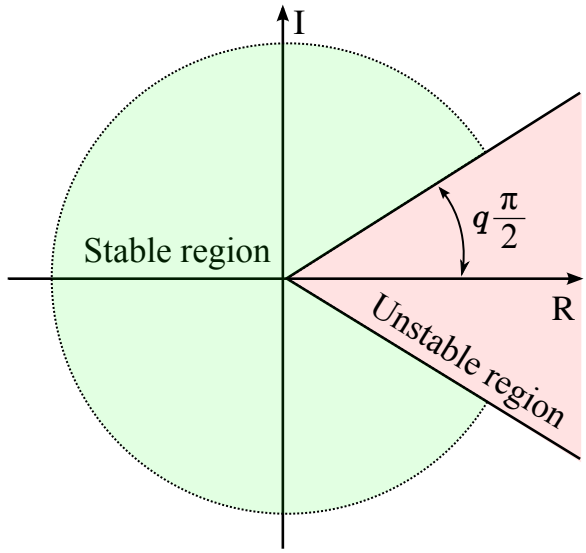


Figure 2.2: LTI fractional-order system stability region for $0 < q < 1$

3. If all obtained roots satisfy the condition (2.1), the system is stable.

Stability regions of a fractional-order system are shown in Figure 2.2.

Note that there are currently no polynomial techniques, either Routh or Jury type, to analyze the stability of fractional-order systems [72].

2.2.3 Time Domain Analysis

Another solution involves numerical computation of fractional-order derivatives which is carried out by means of a revised Grünwald-Letnikov definition (2.2) rewritten as

$$ {}_a\mathscr{D}_t^{\alpha} f(t) = \lim_{h \to 0} \frac{1}{h^{\alpha}} \sum_{j=0}^{\left[\frac{t-a}{h}\right]} w_j^{(\alpha)} f(t - jh), \tag{2.16} $$

where h is the computation step-size and $w_j^{(\alpha)} = (-1)^j \binom{\alpha}{j}$ can be evaluated recursively from

$$ w_0^{(\alpha)} = 1,\ w_j^{(\alpha)} = \left(1 - \frac{\alpha + 1}{j}\right) w_{j-1}^{(\alpha)},\ j = 1,\ 2,\ \cdots . \tag{2.17} $$

To obtain a numerical solution for the equation in (2.9) the signal $\hat{u}(t)$ should be obtained first, using the algorithm in (2.16), where

$$ \hat{u}(t) = b_m \mathscr{D}^{\beta_m} u(t) + b_{m-1} \mathscr{D}^{\beta_m - 1} u(t) + \cdots + b_0 \mathscr{D}^{\beta_0} u(t). \tag{2.18} $$

The time response of the system can then be obtained using the following equation:

$$y(t) = \frac{1}{\sum_{i=0}^{n} \frac{a_i}{h^{\alpha_i}}} \left[u(t) - \sum_{i=0}^{n} \frac{a_i}{h^{\alpha_i}} \sum_{j=1}^{\left[\frac{t-a}{h}\right]} w_j^{(\alpha)} y(t - jh) \right]. \tag{2.19}$$

The presented method is a fixed step method. The accuracy of simulation therefore may depend on the step size [23, 72, 146].

If the system (2.13) has a input-output delay L, the resulting delayed response $y_d(t)$ with $y_d(0) = 0$ is obtained such that

$$y_d(t) = \begin{cases} y(t-L), & t > L \\ 0, & \text{otherwise.} \end{cases} \tag{2.20}$$

2.2.4 Frequency Domain Analysis

Frequency domain response may be obtained by substituting $s = j\omega$ in (2.13). The complex response for a frequency $\omega \in (0; \infty)$ can then be computed as follows:

$$R(j\omega) = \frac{b_m(j\omega)^{\beta_m} + b_{m-1}(j\omega)^{\beta_{m-1}} + \cdots + b_0(j\omega)^{\beta_0}}{a_n(j\omega)^{\alpha_n} + a_{n-1}(j\omega)^{\alpha_{n-1}} + \cdots + a_0(j\omega)^{\alpha_0}} \mathrm{e}^{-L(j\omega)}, \tag{2.21}$$

where j is the imaginary unit.

In addition, consider the following useful relation for the noninteger power $\alpha \in \mathbb{R}$ of the imaginary unit

$$j^{\alpha} = \cos\left(\tfrac{\alpha\pi}{2}\right) + j \sin\left(\tfrac{\alpha\pi}{2}\right). \tag{2.22}$$

2.3 Approximation of Fractional-order Operators

The Oustaloup recursive filter, proposed in [89] and discussed in [72, 137], gives a very good approximation of fractional operators in a specified frequency range. It is a well-established method and is often used for practical implementation of fractional-order systems and controllers. It is summarized next.

In order to approximate a fractional differentiator of order α or a fractional integrator of order $(-\alpha)$ by a conventional transfer function one may compute the zeros and poles of the latter using the following equations:

$$s^{\alpha} \approx K \prod_{k=1}^{N} \frac{s + \omega'_k}{s + \omega_k}, \tag{2.23}$$

where

$$\omega'_k = \omega_b \cdot \omega_u^{(2k-1-\alpha)/N}, \tag{2.24}$$

$$\omega_k = \omega_b \cdot \omega_u^{(2k-1+\alpha)/N}, \tag{2.25}$$

$$K = \omega_h^{\alpha}, \quad \omega_u = \sqrt{\omega_h/\omega_b}, \tag{2.26}$$

and N is the order of approximation in the valid frequency range $(\omega_b; \omega_h)$.

Due to property in (2.5) for fractional orders $\alpha \geq 1$ it holds

$$s^{\alpha} = s^n s^{\gamma}, \tag{2.27}$$

where $n = \alpha - \gamma$ denotes the integer part of α and s^{γ} is obtained by the Oustaloup approximation by using (2.23). Thus, every operator in (2.13) may be approximated using (2.27) and substituted by the obtained approximation, yielding a conventional integer-order transfer function. For digital implementations, the obtained approximation may be converted to its discrete-time equivalent using a suitable method.

2.4 Fractional-order Controllers

The notion of a fractional PID controller was introduced by Podlubny in [98,99]. This generalized controller is called the $\mathrm{PI}^{\lambda}\mathrm{D}^{\mu}$ controller, and has an integrator with an order λ and a differentiator of order μ. Podlubny demonstrated that the fractional-order controller offers superior performance compared to an integer-order one when used in a control loop with a fractional-order plant. In more recent researches [59, 135] it has been confirmed that the fractional controller outperforms the integer-order PID controller.

In the Laplace domain the parallel form of the FOPID controller is given by

$$C_{FOPID}(s) = K_p + K_i s^{-\lambda} + K_d s^{\mu}. \tag{2.28}$$

Obviously, when taking $\lambda = \mu = 1$ the result is the classical integer-order PID controller.

Since we are dealing with band-limited approximations throughout this work, it is important to implement the fractional-order integrator component in (2.28) as

$$G_I(s) = \frac{1}{s^{\lambda}} = \frac{s^{1-\lambda}}{s} \tag{2.29}$$

for $\lambda < 1$ since this ensures the effect of an integer-order integrator at low frequencies thereby resulting in faster convergence of the controlled output to its final value [72].

The fractional lead-lag compensator has the general form [70, 71, 72]

$$C_{LLC}(s) = K \left(\frac{1 + bs}{1 + as} \right)^{\nu}, \tag{2.30}$$

where K, b, a, and ν are design parameters. Assuming $a < b$, for $\nu > 0$ a lead compensator is obtained, otherwise for $\nu < 0$ a lag compensator is obtained.

In this work we also consider fractional-order inversion model based control (FOINVM). In particular, for a system described by (2.14) with a lag $L = 0$ we consider the following compensator:

$$C_{INVM}(s) = \frac{1 + Ts^{\alpha}}{K(1 + T_f s)}, \tag{2.31}$$

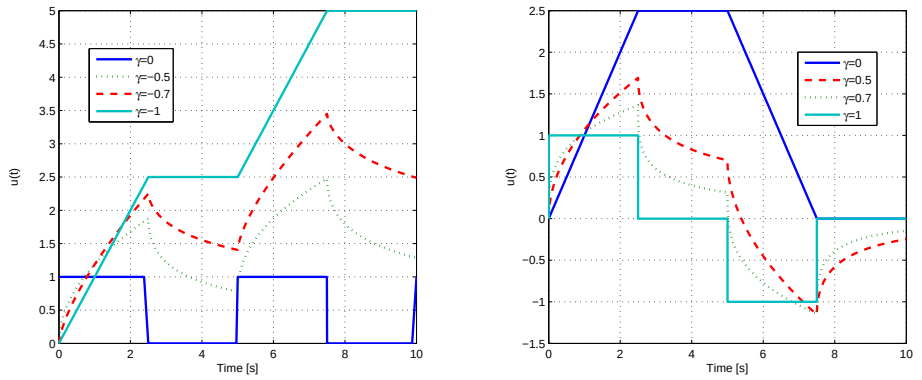
where parameters K, T, and α correspond to those in the FFOPDT model, and T_f is a time constant of a low-pass filter. The latter makes the controller realizable.

Throughout this work we typically assume that the control system is represented by a negative unity feedback of the form

$$G_c(s) = \frac{C(s)G_p(s)}{1 + C(s)G_p(s)}, \tag{2.32}$$

where $C(s)$ is the controller and $G_p(s)$ is the plant under control.

Further we briefly summarize the effects of extending the integral and derivative control actions to the fractional case [72]. The effects of fractional-order integrator and differentiator are shown in Figures 2.3a and 2.3b under square and trapezoidal input signals, respectively.



(a) Fractional integrator

(b) Differentiator

Figure 2.3: Control actions in the time domain corresponding to s^{γ}

The following can be achieved in the frequency domain by varying the power $\gamma \in [-1, 1]$:

- A constant change in the slope of the magnitude curve that varies between -20 dB/dec and 20 dB/dec.

- A constant delay in the phase plot that varies between $-\pi/2$ rad and $\pi/2$ rad.

Consider a comparison between a classical PID controller with unity gains and a fractional one with unity gains and with fractional powers $\lambda = \mu = 0.5$ in the frequency domain given in Figure 2.4.

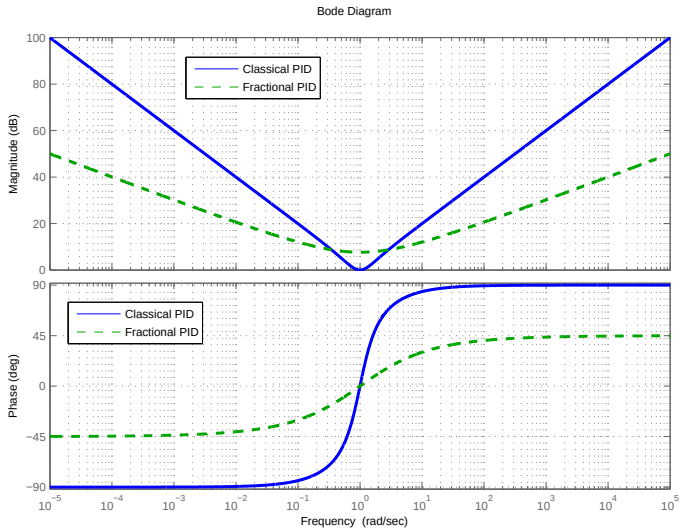


Figure 2.4: Bode diagram of a frequency response of a classical PID controller with $K_p = K_i = K_d = 1$ and a fractional PID controller with $K_p = K_i = K_d = 1$, $\lambda = \mu = 0.5$.

It can be seen that introducing fractional powers for the integral and differential components of a suitable controller has clear benefits due to additional flexibility in tuning of such controllers to meet particular design specifications.

2.5 Optimization Methods

Application of numerical optimization methods form an important part of the present work. Therefore, we provide here the methods employed. The reason for the choice of particular methods is partially based on the reviews in [53, 81] and is detailed in relevant chapters of the present thesis.

2.5.1 Newton-Raphson Method

The Newton-Raphson method is an iterative process which belongs to a class of numerical methods for solving nonlinear equations [7, 45, 49]. We consider

the problem in the form

$$F(x) = 0, \tag{2.33}$$

where $F : \mathbb{R}^n \to \mathbb{R}^n$. Starting from some initial estimate x_0 the next estimate x^+ is obtained by means of

$$x^+ = x + \Delta x, \tag{2.34}$$

where Δx is the solution of

$$J\Delta x = -F, \tag{2.35}$$

and $J = \mathrm{d}F/\mathrm{d}x$ is the Jacobian matrix. In case of an univariate function $f(x)$ this process collapses to the well-known iterative formula

$$x^+ = x - \frac{f(x)}{f'(x)}. \tag{2.36}$$

The Newton-Raphson method is also used for solving subproblems in more sophisticated optimization algorithms.

2.5.2 Nonlinear Least-Squares Estimation Methods

The problem is to obtain a model of a certain system by means of minimization of the sum of squares (residual norm)

$$F = \sum_{i=1}^{n} \varepsilon_i^2 = \|\varepsilon\|_2^2 \,, \tag{2.37}$$

where $\varepsilon_i = y_i - \hat{y}_i$ is the residual (simulation error), y_i is the true system output and $\hat{y}_i$ is the predicted output for collected samples $i = 1, 2, \ldots, N$.

First, we consider a Trust Region Reflective method for handling large-scale bounded problems [25, 73]. Given a trust region Δ_k at every kth iteration the following steps are carried out [15]:

1. Compute F_k, g_k (gradient of F_k), D_k (positive diagonal matrix), H_k and C_k (scaling matrices), define the quadratic model

$$\psi_k(s) = g_k^{\mathrm{T}} s + \frac{1}{2} s^{\mathrm{T}} (H_k + C_k) s. \tag{2.38}$$

2. Compute a step s_k, with $x_k + s_k \in \mathrm{int}(\mathscr{F})$, where $\mathscr{F}$ is the feasible region for search variable values, by solving the subproblem

$$\min_s \left\{ \psi_k(s) : \|D_k s\| < \Delta_k, s \in S_k \right\}, \tag{2.39}$$

where S_k is a small-dimensional subspace in $\mathbb{R}^n$.

3. If $F(x_k + s_k) < F(x_k)$, then $x_{k+1} = x_k + s_k$, otherwise x_k remains unchanged for the next iteration.

4. Adjust the trust region Δ_k.

In case of the least-squares problem the subspace S_k may be determined by taking into account

$$\min_s \left\{ \|Js + F\|_2^2 \right\}, \tag{2.40}$$

where J is the Jacobian of F.

Second, we consider the Levenberg-Marquardt algorithm [64, 74]. The search direction p_k of the Levenberg-Marquardt method is defined by the solution of equations at iteration step k

$$(J_k^{\mathrm{T}} J_k + \lambda_k I) p_k = -J_k^{\mathrm{T}} F_k, \tag{2.41}$$

where J_k is the Jacobian matrix, λ_k is a non-negative scalar, and I is the identify matrix [38].

2.5.3 Nelder-Mead Method

The Nelder-Mead simplex method is used for solving unconstrained optimization problems of the form

$$\min_x F(x), x \in \mathbb{R}^n. \tag{2.42}$$

It is a direct search method, and is therefore well-suited to optimize a function whose derivatives are unknown or non-existent [144]. In the following, we summarize the method described in [53].

First, an initial simplex is constructed by determining $n + 1$ vertices along with corresponding values of F. The kth iteration then consists of the following steps:

1. **Order.** Order the $n + 1$ vertices, so that $F(x_1) \leqslant F(x_2) \leqslant \cdots \leqslant F(x_{n+1})$ is satisfied. Apply tie-breaking rules when necessary.

2. **Reflect.** Compute the reflection point

$$x_r = \bar{x} + \rho(\bar{x} - x_{n+1}), \tag{2.43}$$

where

$$\bar{x} = \sum_{i=1}^{n} x_i / n \tag{2.44}$$

is the centroid of the n best vertices. Evaluate $F_r = F(x_r)$. If $F_1 \leqslant F_r < F_n$, set $x_{n+1} = x_r$ and terminate the iteration.

3. **Expand.** If $F_r < F_1$, calculate the expansion point

$$x_e = \bar{x} + \chi(x_r - \bar{x}), \tag{2.45}$$

and evaluate $F_e = F(x_e)$. If $F_e < F_r$, set $x_{n+1} = x_e$ and terminate the iteration. Otherwise, set $x_{n+1} = x_r$ and terminate the iteration.

4. **Contract.** If $F_r \geqslant F_n$, perform a contraction between $\bar{x}$ and the better of x_{n+1} and x_r:

   (a) **Contract outside.** If $F_n \leqslant F_r < F_{n+1}$, calculate

   $$x_c = \bar{x} + \gamma(x_r - \bar{x}) \tag{2.46}$$

   and evaluate $F_c = F(x_c)$. If $F_c \leqslant F_r$, set $x_{n+1} = x_c$ and terminate the operation. Otherwise, go to Step 5 (perform a shrink).

   (b) **Contract inside.** If $F_r \geqslant F_{n+1}$, perform an inside contraction: calculate

   $$x'_c = \bar{x} - \gamma(\bar{x} - x_{n+1}) \tag{2.47}$$

   and evaluate $F'_c = F(x'_c)$. If $F'_c < F_{n+1}$ set $x_{n+1} = x'_c$ and terminate the iteration. Otherwise, go to Step 5 (perform a shrink).

5. **Shrink.** Define n new vertices from

$$x_i = x_1 + \sigma(x_i - x_1), \quad i = 2, \ldots, n+1, \tag{2.48}$$

and evaluate F at these points.

In the algorithm described above four scalar coefficients are used, i.e., the coefficients of reflection, expansion, contraction, and shrinkage, denoted by ρ, χ, γ, and σ, respectively. According to the original paper [80], these coefficients should satisfy

$$\rho > 0, \quad \chi > 1, \quad 0 < \gamma < 1, \quad 0 < \sigma < 1. \tag{2.49}$$

2.5.4 Optimization Problems with Bounds and Constraints

While certain optimization algorithms, such as the Levenberg-Marquardt and Nelder-Mead methods, are designed to solve unconstrained problems unbounded in the search space, it is possible to introduce both variable search space bounds and constraints [38] in terms of coordinate transformations and penalty functions.

Let x denote the vector of search variables of size $N \times 1$. For bound constraints, a coordinate transformation may be applied to each individual search variable. Let x_i^L and x_i^U denote the lower bound and upper bound

on the ith search parameter, respectively, and denote by z the new search variable vector. Let $\varphi : \mathbb{R} \to \mathbb{R}$ denote a coordinate transformation function, such that $x = \varphi(z)$ and $z = \varphi^{-1}(x)$.

In this work, we consider quadratic and trigonometric transformations. For problems with lower bounds we have

$$x_i = x_i^L + z_i^2, \tag{2.50}$$

from which it follows that $x_i \geqslant x_i^L$, since $z_i^2 \geqslant 0$. Initial estimates $z_{i,0}$ are obtained from original initial estimates $x_{i,0} \geqslant x_i^L$ as

$$z_{i,0} = \sqrt{x_{i,0} - x_i^L}. \tag{2.51}$$

For problems with upper bounds we have

$$x_i = x_i^U - z_i^2, \tag{2.52}$$

from which it follows that $x_i \leqslant x_i^U$, since $-z_i^2 \leqslant 0$. Initial estimates $z_{i,0}$ are obtained from original initial estimates $x_{i,0} \leqslant x_i^U$ as

$$z_{i,0} = \sqrt{x_i^U - x_{i,0}}. \tag{2.53}$$

Finally, if the problem has both lower and upper bounds, we have

$$x_i = x_i^L + (x_i^U - x_i^L)\frac{\sin(z_i) + 1}{2}, \tag{2.54}$$

from which it follows that $x_i^L \leqslant x_i \leqslant x_i^U$, since the values of the function $f(z_i) = (\sin(z_i)+1)/2$ are always inside of the interval $[0, 1]$ and x_i is therefore bounded by $f(z_i) = 0 \Rightarrow x_i = x_i^L$, $f(z_i) = 1 \Rightarrow x_i = x_i^U$. Initial estimates $z_{i,0}$ are obtained from original initial estimates $x_i^L \leqslant x_{i,0} \leqslant x_i^U$ as

$$z_{i,0} = \Re\left(\arcsin\left(\frac{-2x_{i,0} + x_i^L + x_i^U}{x_i^L - x_i^U}\right)\right), \tag{2.55}$$

where $\Re(\cdot)$ denotes the real part.

Next, we consider constrained problems. To introduce constraints, a modification of the cost function

$$\kappa(\cdot) = \kappa^*,$$

where κ^* denotes the cost for the original optimization problem, is necessary. Let us define a function $f_{nz} : \mathbb{R} \to \mathbb{R}_+$ such that

$$f_{nz}(x) := \begin{cases} x, & x > 0 \\ 0, & x \leqslant 0. \end{cases} \tag{2.56}$$

Let us also define a *penalty function* $\rho : \mathbb{R} \to \mathbb{R}$ such that

$$\rho(x) := \begin{cases} \mathrm{e}^{\gamma} - 1 + x, & x > \gamma \\ \mathrm{e}^{x} - 1, & x \leqslant \gamma, \end{cases} \tag{2.57}$$

where $\gamma > 0$ is some predefined constant.

First, for general nonlinear inequality constraints of the form $c^{ni}(\cdot) \leqslant 0$, where $c^{ni} : \mathbb{R}^{q \times r} \to \mathbb{R}^{N_{ni} \times M_{ni}}$, we define the following penalty function $\kappa^{ni} : \mathbb{R}^{N_{ni} \times M_{ni}} \to \mathbb{R}$ as

$$\kappa^{ni}(c^{ni}(\cdot)) := \rho(c^{ni}_{\Sigma}(\cdot)), \tag{2.58}$$

where

$$c^{ni}_{\Sigma}(\cdot) = \sum_{i=1}^{N_{ni}} \sum_{k=1}^{M_{ni}} f_{nz}(c^{ni}_{i,k}(\cdot)). \tag{2.59}$$

Next, for general nonlinear equality constraints of the form $c^{ne}(\cdot) \leqslant 0$, where $c^{ne} : \mathbb{R}^{q \times r} \to \mathbb{R}^{N_{ne} \times M_{ne}}$, we define the penalty function $\kappa^{ne} : \mathbb{R}^{N_{ne} \times M_{ne}} \to \mathbb{R}$ as

$$\kappa^{ne}(c^{ne}(\cdot)) := \rho(c^{ne}_{\Sigma}(\cdot)), \tag{2.60}$$

where

$$c^{ne}_{\Sigma}(\cdot) = \sum_{i=1}^{N_{ne}} \sum_{k=1}^{M_{ne}} f_{nz}(|c^{ne}_{i,k}(\cdot)|), \tag{2.61}$$

where $|\cdot|$ denotes the absolute value.

The complete cost function κ for the constrained optimization problem thereby has the form

$$\kappa = \kappa^{*} + \kappa^{ni} + \kappa^{ne}. \tag{2.62}$$

Chapter 3

Identification of Fractional-order Models

In the following chapter methods for identification of fractional-order models in transfer function form in time and frequency domains are presented and discussed. The chapter is organized as follows. First, system identification fundamentals are presented in Section 3.1. In Section 3.2 the methods proposed for time domain identification of fractional-order models are discussed. In Section 3.3 closed-loop identification methods in the time domain are described. Finally, conclusions are drawn in Section 3.5.

3.1 System Identification Fundamentals

In this work we mostly consider the black box modeling approach [57]. Strictly speaking, no assumptions are made on the internal physical structure of the studied system. The goal of system identification in this case is to infer a dynamic system model based upon experimentally collected data. For a black box model it is necessary to obtain a relationship between system inputs and outputs under external stimuli (input signals, disturbances) in order to determine and predict the system behavior. A general form of a single input-single output system with disturbances is given in Figure 3.1.

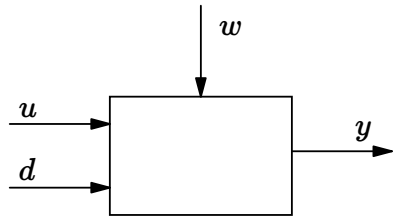


Figure 3.1: A general system with input u, output y, measured disturbance d and unmeasured disturbance w

The general procedure of system identification is summarized in Figure 3.2 and consists of the following stages.

1. Design the experiment. For dynamic systems it is usual to collect transient response data in the time domain by applying a set of predetermined input signals, or frequency response (magnitude and phase) in the frequency domain (e.g. by doing a frequency sweep).

2. Record the dataset based on an experiment. The collected data must be as informative as possible subject to potential constraints.

3. Choose a set of models and/or the model structure and the criterion to fit.

4. Calculate the model using a suitable algorithm.

5. Validate the obtained model. It is desirable to use two different datasets for identification and validation.

6. If the model is satisfactory, use it for the desired purpose. Otherwise, revise modeling/identification strategy and repeat the above steps.

A critical step in the identification process is the determination of the amount of contribution of noise and disturbances to the collected data.

In what follows, we investigate methods for system identification by a fractional-order model based on time domain and frequency domain experiments.

3.2 Open-loop Identification in the Time Domain

Suppose that experimental data is collected from a general single input, single output nonlinear system $\Psi : \mathscr{I} \to \mathscr{O}$, where $(\mathscr{I}, \mathscr{O}) \subset \mathbb{R}^2$ denote the measured input and output signals, respectively, such that

$$z(t) = \Psi(v(t)) + \mathfrak{N}, \tag{3.1}$$

where $z(t)$ denotes the system output, and $v(t)$ denotes the system input, and $\mathfrak{N}$ denotes measurement noise, and is represented by a data set holding the samples from the system input $u_k = v(kt_s)$ and output $y_k = z(kt_s) + \mathfrak{N}$ under a uniform sample rate $t_s = t_{k+1} - t_k \equiv \text{const}$:

$$Z_N = \{u_0, y_0, u_1, y_1, \ldots, u_N, y_N, t_s\}, \tag{3.2}$$

where $k = 0, 1, \ldots, N$. Since zero initial conditions are assumed, then if $z(0) = y_0 \neq 0$, then the offset is removed from each of the collected output samples by means of

$$y_k = y_k - y_0, \quad k = 0, 1, \ldots, N. \tag{3.3}$$

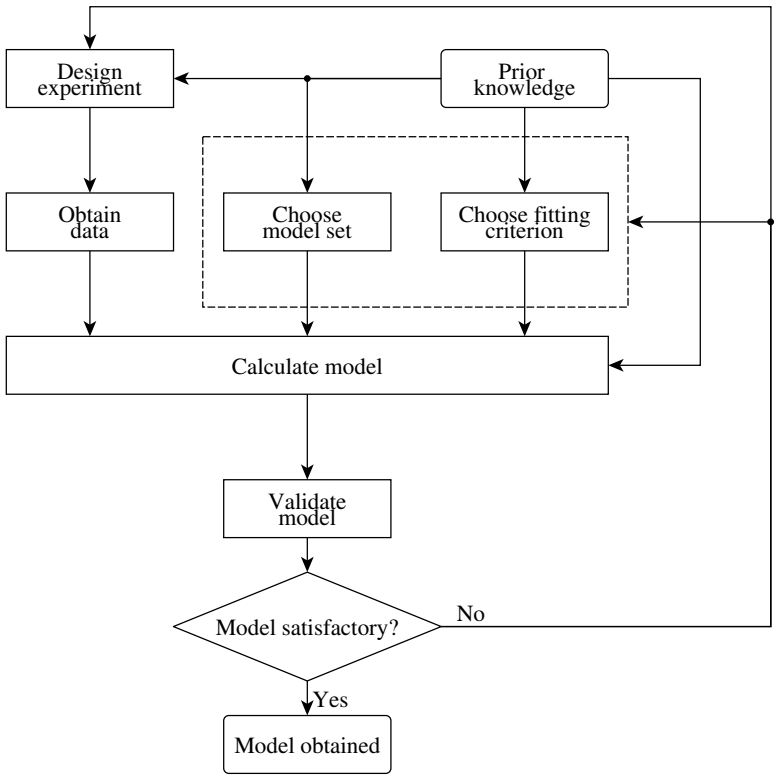


Figure 3.2: Typical system identification procedure

In case of fractional-order systems the key issue is the evaluation of the response of the model [24, 50]. For the purpose of time domain simulation two methods are considered:

1. Grünwald-Letnikov definition based numerical solver described in Section 2.2.3. Since the whole history of differentiation is preserved, this method is considered the most accurate. The amount of necessary computations grows exponentially with N, since the method considers the whole history of computation; therefore, the identification process may become slow.

2. Oustaloup recursive filter approximation described in Section 2.3 and conventional time domain simulation methods. The amount of necessary computations grows linearly with N, since only a limited history of the process is considered.

The identification method is based on the minimization of model output error in the least-square sense. The corresponding problem is stated as

$$\min_{\theta} \varepsilon^2, \tag{3.4}$$

where $\varepsilon = y_k - \hat{y}_k$ and $\hat{y}_k = \hat{\Psi}(u_k, \theta)$ denotes the response of the estimated system model $\hat{\Psi}$ under the input signal u_k, $k = 0, \ldots, N$, and θ denotes the estimated parameters of the model.

In this thesis, the problem of identification of transfer function system models is investigated. The proposed identification parameter selection approach combines the ideas in [50, 62, 63]. In addition, we include estimation of the input-output delay. Recall the model from (2.13)

$$G(s) = \frac{b_m s^{\beta_m} + b_{m-1} s^{\beta_{m-1}} + \cdots + b_0 s^{\beta_0}}{a_n s^{\alpha_n} + a_{n-1} s^{\alpha_{n-1}} + \cdots + a_0 s^{\alpha_0}} \mathrm{e}^{-Ls}. \tag{3.5}$$

The identification problem is hereinafter stated as a problem of estimating a set of parameters $\theta = \begin{bmatrix} \theta_m & L \end{bmatrix}$ of the model in (3.5), where θ_m is formed by

$$\begin{aligned} a_p &= \begin{bmatrix} a_n & a_{n-1} & \cdots & a_0 \end{bmatrix}, \quad \alpha_p = \begin{bmatrix} \alpha_n & \alpha_{n-1} & \cdots & \alpha_0 \end{bmatrix}, \\ b_z &= \begin{bmatrix} b_m & b_{m-1} & \cdots & b_0 \end{bmatrix}, \quad \beta_z = \begin{bmatrix} \beta_n & \beta_{n-1} & \cdots & \beta_0 \end{bmatrix}, \end{aligned} \tag{3.6}$$

a_p and b_z denote pole and zero polynomial differential operator coefficients, α_p and β_z denote the corresponding exponents (orders of differentiation), respectively; if $\alpha_0 = \beta_0 = 0$, then the system static gain is identified as $K = b_0/a_0$; and for θ_m there exists 9 possible parameter sets depending on the chosen identification method.

- Full model parameter identification, $\theta_m = \begin{bmatrix} a_p & \alpha_p & b_z & \beta_z \end{bmatrix}$;
- Fix orders, identify coefficients, $\theta_m = \begin{bmatrix} a_p & b_z \end{bmatrix}$;
- Fix coefficients, identify orders, $\theta_m = \begin{bmatrix} \alpha_p & \beta_z \end{bmatrix}$.

With pole polynomial fixed:

- Identify zero polynomial coefficients and orders, $\theta_m = \begin{bmatrix} b_p & \beta_z \end{bmatrix}$,
- Fix orders, identify zero polynomial coefficients, $\theta_m = b_z$,
- Fix coefficients, identify zero polynomial orders, $\theta_m = \beta_z$.

With zero polynomial fixed:

- Identify pole polynomial coefficients and orders, $\theta_m = \begin{bmatrix} a_p & \alpha_z \end{bmatrix}$,
- Fix orders, identify pole polynomial coefficients, $\theta_m = a_z$,
- Fix coefficients, identify pole polynomial orders, $\theta_m = \alpha_z$.

The choice of the particular set θ_m depends on the following.

- If the system under investigation is of commensurate order $0 < \gamma < 2$, an initial guess model should be generated

$$\alpha_p = [\ \gamma n \quad \gamma(n-1) \quad \cdots \quad 0\ ] \tag{3.7}$$

and

$$\beta_z = [\ \gamma m \quad \gamma(m-1) \quad \cdots \quad 0\ ], \tag{3.8}$$

such that

$$\left\{(n,m) \in \mathbb{Z}_+^2 : n \geqslant m\right\}, \tag{3.9}$$

where n determines the pseudo-order of the system; the orders should be fixed and only model coefficients are estimated; A commensurate initial model may also be used for identification of all parameters;

- If either zero or pole polynomials are known, they may be fixed;
- In general, if there is a considerable amount of parameters to identify, the identification process may be slow—fixing parameters at particular values allows reducing the amount of necessary computations.

The following two algorithms, discussed in Subsection 2.5.2, are used for nonlinear least-squares estimation of model parameters θ:

1. Trust Region Reflective algorithm—this method handles search variable bounds and is intended for solving large-scale problems [15]. Bounds are given for parameter sets as $\theta_b = \{\theta_{min}, \theta_{max}\}$.
2. Levenberg-Marquardt algorithm—the method is quite robust and is commonly used for black-box model identification [26, 150]. However, the conventional implementation does not handle bound constraints. Therefore, to correctly identify model orders $\alpha_p \geqslant 0$ and $\beta_z \geqslant 0$ and the delay $L \geqslant 0$ a coordinate transform is required, as discussed in Subsection 2.5.4. In the new coordinates z_j the corresponding parameters are given as

$$\theta_j = z_j^2, \quad j = 0, 1, 2, 3 \ldots . \tag{3.10}$$

In what follows, only continuous-time system identification is employed. However, real systems are sampled with a finite sample rate t_s. According to the Nyquist-Shannon sampling theorem [57], the information about frequencies f_Ψ, such that

$$f_\Psi > \frac{f_s}{2} = \frac{1}{2t_s}, \tag{3.11}$$

where f_s is the sampling frequency, is lost. This should be taken into account when studying the behavior of the identified model and, consequently, in model based control design applications. Moreover, high-frequency components in the experimental signal may result in unwanted effects, such as aliasing. To prevent this, filtering of the collected signal data is necessary, and is usually accomplished by means of a stationary low-pass filter [63].

3.2.1 Parametric Identification

A separate model parameter estimation problem is the so-called parametric identification. In the context of this thesis, we consider the following model structure

$$G(s) = \frac{Z(p,q)}{P(p,q)} D(p), \tag{3.12}$$

where $Z(p,q)$ is the zero polynomial, $P(p,q)$ is the pole polynomial, and $D(p)$ is the delay term, parametrized such that

$$\theta = \{p_1, p_2, \dots, q_1, q_2, \dots\}, \tag{3.13}$$

where some of the parameters are known *a priori*. Parametric identification is thus very useful if the structure of the model is precisely known, e.g., in case of a closed control loop. This will be further discussed in Section 3.3. The procedure for parameter estimation is the same as discussed in the previous section.

3.2.2 Residual Analysis

The following discussion is related to the assessment of the quality of the identified model.

Denote by y_r the experimental plant output, and by y_m the identified model output. We consider the SISO case, so both y_r and y_m should be vectors of size $N \times 1$. In the following, we address the problem of statistical analysis of modeling residuals. Analysis is partially due to Ljung [57]. Residuals are given by a vector containing the model output error

$$\varepsilon = y_r - y_m. \tag{3.14}$$

The percentage fit may be expressed as

$$Fit = \left(1 - \frac{\|\varepsilon\|}{\|y_r - \bar{y}_r\|}\right) \cdot 100\%, \tag{3.15}$$

where $\|\cdot\|$ is the Euclidean norm, and $\bar{y}_r$ denotes the mean value of y_r.

Basic statistical data may be computed first, such as maximum absolute error

$$\varepsilon_{max} = \max_k |\varepsilon(k)|, \tag{3.16}$$

and mean squared error

$$\varepsilon_{MSE} = \frac{1}{N} \sum_{k=1}^{N} \varepsilon_k^2 = \frac{\|\varepsilon\|_2^2}{N}. \tag{3.17}$$

Additional useful information is given by an estimate for autocorrelation of residuals for lag $\tau = 1, 2, \ldots, \tau_{max} < N$, which may be computed by means of

$$R_\varepsilon(\tau) = \frac{1}{(N-\tau)} \sum_{k=1}^{N-\tau} \varepsilon(k)\varepsilon(k+\tau). \tag{3.18}$$

The vector $r^\varepsilon = \begin{bmatrix} R_\varepsilon(0) & R_\varepsilon(1) & \cdots & R_\varepsilon(\tau_{max}) \end{bmatrix}$ is constructed and is normalized such that $r^{\varepsilon,norm} = r^\varepsilon / R_\varepsilon(0)$. Assuming normal distribution of residuals the confidence band $\hat{\eta}$ is then approximated for a confidence percentage $p_{conf} \in (0, 1]$ around zero mean as an interval

$$\hat{\eta} = \left[\frac{0 - \Phi^{-1}(c_p)}{\sqrt{N}}, \frac{0 + \Phi^{-1}(c_p)}{\sqrt{N}} \right], \tag{3.19}$$

where $c_p = 1 - 0.5(1 - p_{conf})$ and $\Phi^{-1}(x) = \sqrt{2}\,\mathrm{erf}^{-1}(2x - 1)$ is the quantile function.

Based on these data, the following conclusions may be made about the quality of the model:

- Maximum absolute error ε_{max} shows the maximum deviation from the expected behavior of the model over the examined time interval, however, it may be misleading in case of disturbances or strong noise;
- The mean squared error ε_{MSE} may serve as a general measure of model quality. The lower it is, the more likely the model represents an adequate description of the studied process;
- If the residual samples represent an uncorrelated white noise process, then the following condition should ideally hold:

$$r_i^{\varepsilon,norm} \in \hat{\eta} \quad \forall i = 1, 2, \ldots, \tau_{max}. \tag{3.20}$$

Consider now an illustrative example.

Example 3.1 Identification data is collected from a system

$$\Psi = \Psi_G + \mathfrak{N}, \tag{3.21}$$

where Ψ_G is given by a continuous-time fractional-order transfer function of the form

$$\Psi_G(s) = \frac{1.5}{0.11s^{1.93} + 0.79s^{0.31} + 1}, \tag{3.22}$$

and the noise term has an amplitude of $\mathfrak{N} = \pm 0.05$. A pseudo-random binary sequence is used as the excitation signal for obtaining the transient response with a sample time of 0.01s. The corresponding experimental dataset Z_G is depicted in Figure 3.3.

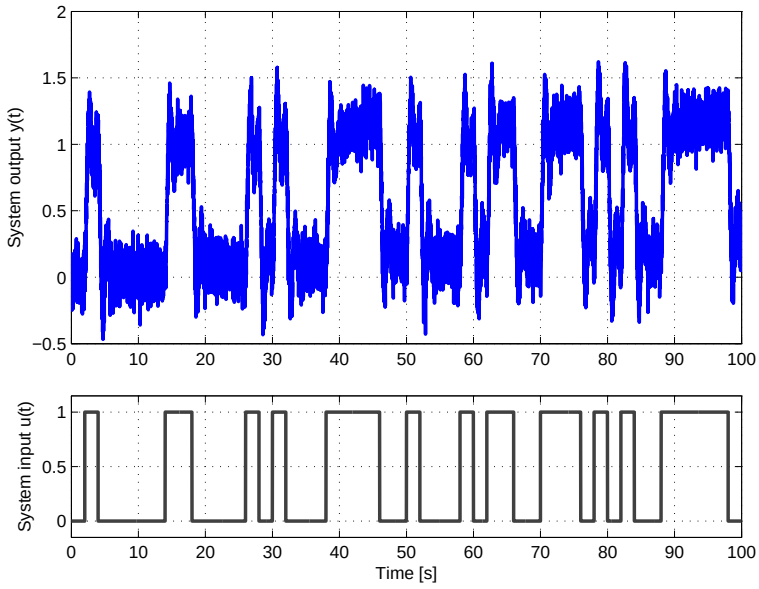


Figure 3.3: Example system identification dataset

The parameter estimation methods are applied to the problem, where the initial model is generated with various commensurate orders $\gamma = \{0.5, 1.0, 1.5\}$ and a pseudo-order $n = 2$, and has the form

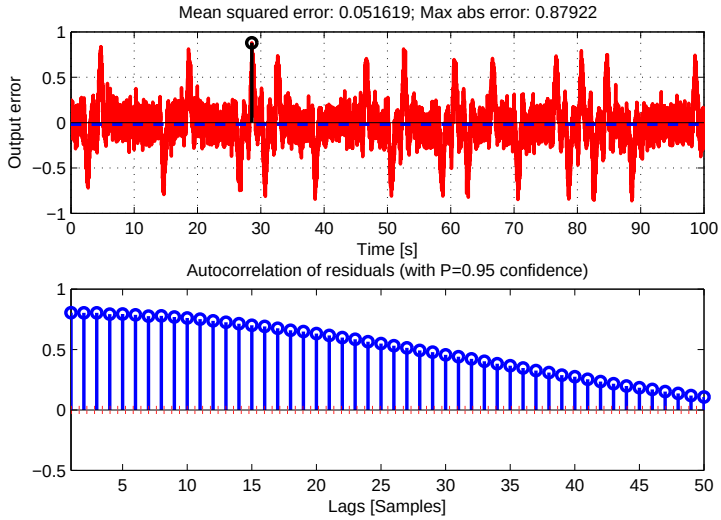
$$G_0(s) = \frac{1}{s^{2\gamma} + s^{\gamma} + 1}. \tag{3.23}$$

Assuming we have no knowledge of the system other than its pseudo-order, we choose to identify all of the parameters of the model. The identification is carried out using Trust Region Reflective and Levenberg-Marquardt algorithms, the latter initialized with $\lambda = 0.01$. In both cases, the problem is unbounded in search variables. The systems are simulated using the Grünwald-Letnikov definition based solver. The algorithms are implemented in MATLAB computer software as detailed in Chapter 6. The results of identification are presented in Table 3.1. In three cases the algorithms converged yielding the same result

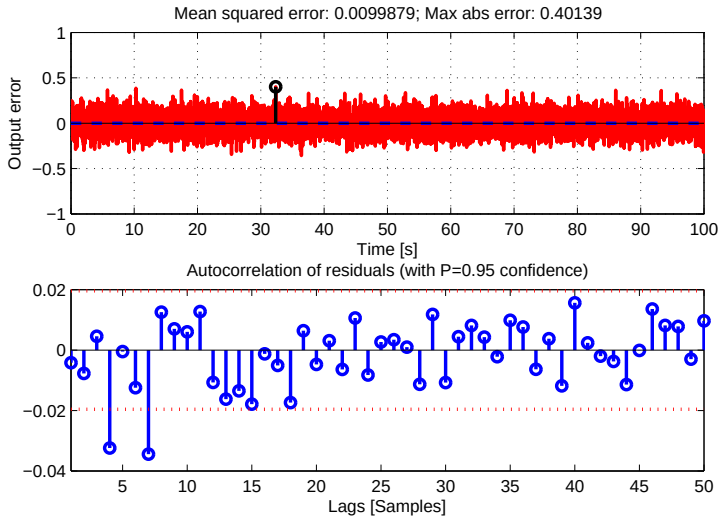
$$\hat{\Psi}_G(s) = \frac{1.5016}{0.11027s^{1.9271} + 0.78844s^{0.30846} + 1}. \tag{3.24}$$

Overall, the LM algorithm appears to be more robust in this case. It only failed to converge once, while the TRR method failed to converge twice.

Comparative residual analysis for cases $\gamma = 1.0$ is presented in Figure 3.4. It can be seen that if residuals are formed by the noise term $\mathfrak{N}$, the autocor-



(a) Trust-Region-Reflective algorithm



(b) Levenberg-Marquardt algorithm

Figure 3.4: Residual analysis for the system identified using nonlinear least squares estimation algorithms with $\gamma = 1.0$

Table 3.1: Comparison of the nonlinear least squares estimation algorithms

Algorithm	γ	%Fit	ε_{MSE}	NoIter	FunEval	τ, min
TRR	0.5	80.42	$9.98 \cdot 10^{-3}$	18	114	01:34
LM	0.5	80.42	$9.98 \cdot 10^{-3}$	15	105	01:26
TRR†	1.0	55.48	$51.62 \cdot 10^{-3}$	83	504	06:55
LM	1.0	80.42	$9.98 \cdot 10^{-3}$	11	79	01:05
TRR‡	1.5	8.94	0.21598	78	474	06:33
LM*	1.5	–	0.627	2	18	00:16

Notes
† failed to converge in 500 function evaluations
‡ convergence to a local minimum
∗ failed to converge, identified system unstable

relation coefficients under different lags generally fall inside of the confidence interval.

Further examples are provided in Chapters 6 and 7.

3.3 Closed-loop Identification in the Time Domain

In this work, we consider the closed-loop plant model identification problem [117]. The structure of the control system is depicted in Figure 3.5. We investigate two identification approaches:

- The indirect approach. The controller is assumed to be known. The identification data set is given by

$$Z_N^i = \{r_0, y_0, r_1, y_1, \ldots, r_N, r_N, t_s\}, \tag{3.25}$$

  where r_k and y_k denote the reference signal (set point) and plant output, respectively, collected at points r and y in Figure 3.5. In this work, we investigate the problem of identification of a closed-loop model in (2.32), where the parameters of $C(s)$ are known. The model structure of the plant $G(s)$ can be easily reconstructed, once the parameters of the closed-loop system are obtained.

- The direct approach. The feedback is ignored and open-loop identification is employed. The experimental data set used for identification is given by

$$Z_N^d = \{u_0, y_0, u_1, y_1, \ldots, u_N, y_N, t_s\}, \tag{3.26}$$

where u_k and y_k denote the plant input and output signal samples collected at points u and y in Figure 3.5, and $k = 1, 2, \ldots, N$. The structure of the model to be identified is explicitly given by (3.5).

If one were to assume that the controller in Figure 3.5 is linear, time-invariant and noise free, then the two methods are equivalent [34]. However, in industrial practice, such a controller is rarely realizable due to, e.g., actuator saturation and measurement noise. Therefore, the selection of the identification approach depends on the availability of necessary measurements.

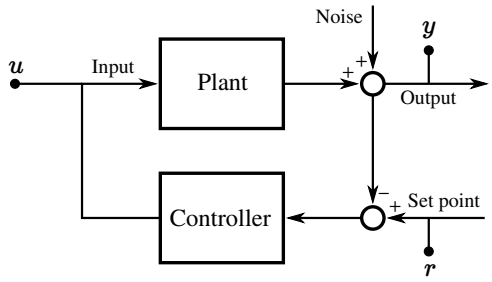


Figure 3.5: Closed-loop control system structure

To carry out the identification in case of known parameters we use the parametric identification method. Consider an example, which illustrates the parametrization approach.

Example 3.2 Let us assume that a closed-loop control system of the form (2.32) is studied and only information about the reference and output values is available. Then, the indirect identification approach must be used. Also, the plant is known to have the structure

$$G_p(s) = \frac{K}{a_2 s^{\alpha_2} + a_1 s^{\alpha_1} + a_0}. \tag{3.27}$$

The closed-loop transfer function with a controller of the form (2.28) is given by

$$G_{cl}(s) = \frac{K G_{pid}(s)}{s(a_2 s^{\alpha_2} + a_1 s^{\alpha_1} + a_0) + K G_{pid}(s)}, \tag{3.28}$$

where $G_{pid}(s) = \left(K_p s + K_i s^{1-\lambda} + K_d s^{1+\mu}\right)$ and the parameters of the FOPID controller are assumed to be known. Then, the model in (3.28) may be parametrized as

$$G_{cl}(s) = \frac{p_1 G_{pid}(s)}{s(p_2 s^{q_1} + p_3 s^{q_2} + p_4) + p_1 G_{pid}(s)}, \tag{3.29}$$

and the identification parameter set is given by

$$\theta = \{p_1, p_2, p_3, p_4, q_1, q_2\}. \tag{3.30}$$

A particular example is provided in Chapter 6.

3.4 Frequency domain Identification in Automatic Tuning Applications for Process Control

In this section we investigate the problem of online identification of a dynamical model of a process, given by (2.14), and assuming that the process under investigation exhibits low-order dynamics and is not of integrating type [149]. Recall that the this model is given by

$$G(s) = \frac{K}{1 + Ts^{\alpha}} \mathrm{e}^{-Ls}, \tag{3.31}$$

where K is the static gain, L is the delay, T is the time constant, and α is the fractional operator order, whereby the essential dynamics of the model are governed by the two latter parameters.

Parameter estimation for autotuning applications is often done by means of frequency domain methods. In particular, relay feedback based experiments are often considered [4, 6, 149]. The controlled system output is in steady state and will only deviate from it slightly. This allows to use relay feedback to tune the controller without significantly changing the set point of the working industrial system thereby preventing possible production losses. In this work, we consider an updated relay feedback approach [72], which is illustrated in Figure 3.6. According to [149], relay feedback based identification is more efficient than step response methods for when the normalized dead time $\tau_n = L/T$ is such that

$$\tau_n < 0.28 \tag{3.32}$$

with temperature and composition control loops in process industries generally satisfying this condition. Note that this applies to conventional FOPDT systems, so in this work we treat this condition as an approximation.

The relay feedback is summarized next for the case, when the relay delay is $d = 0$. First, the static gain of the model K in (3.31) can be obtained by means of

$$K = \frac{\Delta y}{\Delta u}, \tag{3.33}$$

where Δy is the observed change in system output due to change in input Δu; the lag L is determined from

$$L = t_u - t_r, \tag{3.34}$$

where t_u is the time instance, when a new control input is applied to the system, and t_r is the time instance, when a corresponding change in system output is observed.

Next, let

$$\delta = \{(1 - \gamma)u_s, (1 + \gamma)u_s\}, \tag{3.35}$$

where u_s is the current steady-state control law, and $\gamma \in (0, 1)$, and apply a periodic control signal to the plant input, such that $u(t) = \pm\delta$. The switching properties are such that when the relay feedback system senses a change in the direction of change of the process output, the input switches such that $u(t) \leftarrow -u(t)$. After several periods a limit cycle oscillation occurs in most processes. It is then possible to measure the ultimate period T_u and compute the ultimate frequency ω_u and ultimate gain A_u as follows

$$\omega_u = \frac{2\pi}{T_u}, \quad A_u = \frac{4\delta}{\pi a}, \tag{3.36}$$

where δ is the relay amplitude and a is the process amplitude. This information is enough to estimate a conventional FOPDT model, where $\alpha = 1$ in (3.31).

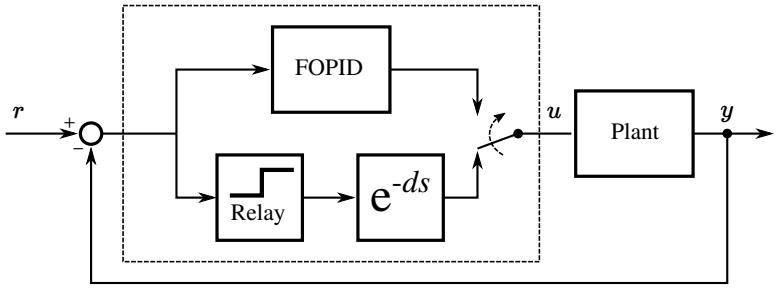


Figure 3.6: PID control with relay test for model parameter estimation

For identification of a FFOPDT model at least two points are required to determine the slope of the magnitude curve. By changing the delay of the relay d it is possible to identify more points. To determine the value of d for a particular frequency of interest ω_d, the following approximation may be used:

$$d \approx \frac{2\pi - \omega_d T}{4\omega_d}, \tag{3.37}$$

assuming an estimation of the time constant T is available. The frequency response points may be distributed logarithmically, so N_f frequencies at which the response is sought may be computed using

$$\omega_{k+1} = \phi^k \omega_k, \quad \omega_0 = \omega_u, \quad k = 0, 1, \ldots, N_f, \tag{3.38}$$

where $0 < \phi < 1$ is some suitable factor. Once several points are collected, identification is done on the basis of the following discussion.

The gain $A(j\omega)$ of (3.31) at a particular frequency ω may be precisely computed as

$$A(j\omega) = |G(j\omega)| = \frac{|K|}{\sqrt{1 + 2\cos(\alpha\pi/2)\omega^\alpha T + \omega^{2\alpha} T^2}}. \tag{3.39}$$

Solving this equation for T, assuming $\omega > 0$, $T > 0$ and $0 < \alpha \leqslant 1$ yields

$$T = \frac{\sqrt{K^2 - \sin^2\left(\frac{\alpha\pi}{2}\right) A^2} - A\cos\left(\frac{\alpha\pi}{2}\right)}{A\omega^{\alpha}} \quad \forall\omega, \tag{3.40}$$

where ω and A correspond to a point on the frequency response curve; for $1 < \alpha < 2$ the solution is given by

$$T = \begin{cases} \dfrac{-\sqrt{K^2 - \sin^2\left(\frac{\alpha\pi}{2}\right) A^2} - A\cos\left(\frac{\alpha\pi}{2}\right)}{A\omega^{\alpha}} & \text{if } \omega \leqslant \omega_p, \\ \dfrac{\sqrt{K^2 - \sin^2\left(\frac{\alpha\pi}{2}\right) A^2} - A\cos\left(\frac{\alpha\pi}{2}\right)}{A\omega^{\alpha}} & \text{if } \omega > \omega_p, \end{cases} \tag{3.41}$$

where

$$\omega_p = \left(\frac{\cos\left(\pi - \frac{\alpha\pi}{2}\right)}{T}\right)^{\frac{1}{\alpha}}. \tag{3.42}$$

Based on this information we can now propose a simple search method to estimate both parameters α and T. Note that this method is tailored to the particular problem. A more general approach may be considered for more complex models [44, 129].

The optimization problem is formulated as

$$\min_{\alpha} J(\alpha), \tag{3.43}$$

where $J(\alpha)$ is a cost function computed for a particular value of α as follows

$$J(\alpha) = \sum_{k=1}^{M} \left(A(j\omega_k) - \tilde{A}_k\right)^2, \tag{3.44}$$

where $(\omega_k, \tilde{A}_k)$ is an identified frequency response point. An average $\bar{T}$ of the time constant is computed using all T_k, each computed for the identified point. Essentially, this is a least-squares optimization problem. The following proposition establishes the conditions on choosing the points on the frequency response curve.

Proposition 3.1 *Denote by*

$$(\omega_\star, A_\star) \in \mathbb{R}^2_+ \tag{3.45}$$

a pair of points such that $A_\star = |G(j\omega_\star)|$, where $G(\cdot)$ is the model in (3.31) *with a static gain K. If there holds*

$$A_\star \leqslant |K|, \tag{3.46}$$

then T may be uniquely determined by (3.40).

Proof: From (3.40), (3.41), and (3.42) it follows that T may be computed using (3.40) in the following cases

- for $0 < \alpha \leqslant 1$ for any $\omega > 0$,
- for $1 < \alpha < 2$, if the following condition holds:

$$\omega > \left(\frac{\cos\left(\pi - \frac{\alpha\pi}{2}\right)}{T} \right)^{\frac{1}{\alpha}}. \tag{3.47}$$

Next, we solve

$$|G(j\omega)| \leqslant |K| \tag{3.48}$$

for ω and obtain

$$\omega \geqslant \left(\frac{2\cos\left(\pi - \frac{\alpha\pi}{2}\right)}{T} \right)^{\frac{1}{\alpha}}, \tag{3.49}$$

which holds for $1 < \alpha < 2$. Combining the above results it is found that for the complete interval $\alpha \in (0, 2)$ if it holds

$$\left(\frac{\cos\left(\pi - \frac{\alpha\pi}{2}\right)}{T} \right)^{\frac{1}{\alpha}} < \left(\frac{2\cos\left(\pi - \frac{\alpha\pi}{2}\right)}{T} \right)^{\frac{1}{\alpha}}, \tag{3.50}$$

then also

$$A_\star \leqslant |K| \tag{3.51}$$

and T may be uniquely determined using (3.40). ■

The above proposition allows us to choose only such points subject to a condition based on known information, since initially we do not know neither α nor T. In the following, the estimation algorithm based on a direct search method is summarized.

1. Given frequency response points $(\omega, A) \in \Omega_{fr} \subset \mathbb{R}^2_+$ with cardinality $|\Omega_{fr}| = M$, choose a set $\Omega_h \in \mathbb{R}^+$ with cardinality $|\Omega_h| = N$ with search step sizes such that $\forall h_i \in \Omega_h : h_i < h_{i+1}, i = 1, 2, \ldots N$;

2. Start with $\alpha = 1$ and $i = 1$. Find the direction of change in the cost function that is determine the minimum

$$\min\{J(\alpha - h_i); J(\alpha); J(\alpha + h_i)\}, \tag{3.52}$$

where $J(\alpha)$ is computed using (3.44) using all available values in Ω_{fr}. If $J(\alpha)$ is the minimal value, stop the estimation process, return $\alpha = 1$ and compute the average $\bar{T}$ using (3.40) from all entries in Ω_{fr}. Otherwise, determine the sign σ as follows and proceed to the next step.

(a) If $J(\alpha - h_1) \leqslant J(\alpha + h_1)$, then $\sigma = -1$;

(b) If $J(\alpha - h_1) > J(\alpha + h_1)$, then $\sigma = 1$.

3. Update α such that $\alpha \leftarrow \alpha + \sigma k h_i$, $k = 1, 2, \ldots, \left\lfloor \frac{1-h_i}{h_i} \right\rfloor$, where $\lfloor \cdot \rfloor$ is the floor function, until a condition $J(\alpha + \sigma(k-1)h_i) < J(\alpha + \sigma k h_i)$ is detected; when the condition occurs:

   (a) If $i > 1$, then re-center α such that $\alpha \leftarrow \alpha - \sigma(h_{i-1} + h_i)$;

   (b) Set $i \leftarrow i + 1$.

   (c) Repeat step 3, while $i \leqslant N$. Otherwise, proceed to step 4.

4. Return the minimizer α. Compute $\bar{T}$ using (3.40) from all values of Ω_{fr}.

If the algorithm fails to converge, then it is possible that the model cannot be determined from the collected frequency response points. In such a case it is either necessary to collect more points, or to use a different identification approach.

In this work we use the following particular step sizes $\Omega_h = \{0.25, 0.1, 0.01\}$. Consider an example that verifies the validity of the above algorithm.

Example 3.3 Consider several FFOPDT systems in (3.31) with nominal lag $L = 1$ s and described by the following sets of parameters. The gains are given by

$$K = \{-5.00, 0.75, 10\}, \tag{3.53}$$

the time constants by

$$T = \{5.00, 25.00, 120.00\}, \tag{3.54}$$

and operator orders by

$$\alpha = \{0.19, 0.81, 1.00, 1.57\}. \tag{3.55}$$

The chosen parameters should provide sufficient coverage for the purpose of initial validation the proposed identification algorithm.

The magnitude response of these models is studied: the first point collected corresponds to ultimate frequency and gain (ω_u, A_u) and further points are collected for frequencies computed as

$$\omega_{k+1} = \phi_r \omega_k, \quad \omega_0 = \omega_u, \quad k = 0, 1, 2, \ldots 6, \tag{3.56}$$

where $\phi_r = 0.5$. The computed magnitude A_k has an uncertainty of $\pm 20\%$, simulating this way real-life measurements. True values of system gain K and lag L are assumed to be known and are used in the computations. In

addition, if $A_k \geqslant K$, it is discarded from the identification dataset. The points are then plugged into the estimation algorithm. The fit to the original model is computed using (3.15), where the residual vector ε is formed using 100 points of magnitude response of the original and identified models. The results can be seen in Tables 3.2 through 3.5. In the tables, $\tilde{\alpha}$ and $\tilde{T}$ denote the estimated values of α and T, respectively. In addition, the number of frequency response points considered for the estimation procedure, as well as the number of algorithm steps necessary to obtain the solution, are given.

The obtained results indicate the validity of the algorithm, since in most cases the essential dynamics of the model are recovered under a relatively large measurement uncertainty, and the resulting models may be used for control design. It was found that the results of parameter estimation depend on the choice of ϕ_r such that $\phi_r \propto \alpha$. The algorithm behaves less reliably with $\alpha > 1$. On four occasions with $\alpha = 1.57$ the estimation algorithm failed to converge completely. This can be remedied for that particular value of α by choosing $\phi_r = 0.8$.

As expected, under 0% uncertainty the algorithm converges to ideal solutions in all studied cases indicating the validity of the identification algorithm.

3.5 Conclusions

In this chapter, problems related to identification of single input, single output fractional-order transfer function models were discussed. In particular, methods were proposed for time domain identification of both commensurate and incommensurate fractional-order systems; parametric identification for estimating unknown parameters, including closed-loop identification problems, where the parameters of the control system are partially known; frequency domain identification of FFOPDT plants for automatic tuning applications. The differences with respect to the state of the art are as follows. The time domain identification procedure focuses on parametric identification, which allows expanding it to solving a wide set of problems, e.g., closed-loop identification. The particular implementation thereof of it is different from alternative solutions, in that it is more flexible in forming the search variable space. The described identification algorithms form an essential part of a bigger framework [112]. The proposed frequency domain identification method also belongs to larger set of tools aimed at implementing automatic tuning of industrial FOPID controllers. The advantages of this method is that it is based on direct search, so it is computationally more stable than gradient-based methods. There is also no need to solve matrix equations as in the case of least-squares based frequency domain identification methods. The proposed method can be conveniently implemented and used in embedded applications.

Table 3.2: Frequency domain identification algorithm validation for the FFOPDT plant with $\alpha = 0.19$

K	T	ω_u	A_u	$\tilde{\alpha}$	$\tilde{T}$	%Fit	Points	Steps
-5.00	5.00	2.885	0.7065	0.23	5.03	71.53	6	15
0.75	5.00	2.885	0.106	0.14	5.01	63.17	6	15
10.00	5.00	21.72	1.007	0.17	5.07	89.08	6	21
-5.00	25.00	2.852	0.1589	0.20	25.09	93.61	6	18
0.75	25.00	2.852	0.02383	0.21	26.09	84.84	6	17
10.00	25.00	2.852	0.3178	0.20	24.29	78.16	6	18
-5.00	120.00	2.845	0.03394	0.14	124.10	50.24	6	15
0.75	120.00	2.845	0.005091	0.13	121.26	52.71	6	16
10.00	120.00	2.845	0.06788	0.15	114.72	79.06	6	14

Table 3.3: Frequency domain identification algorithm validation for the FFOPDT plant with $\alpha = 0.81$

K	T	ω_u	A_u	$\tilde{\alpha}$	$\tilde{T}$	%Fit	Points	Steps
-5.00	5.00	1.975	0.5541	0.69	4.62	73.34	6	17
0.75	5.00	1.975	0.08312	0.74	5.01	79.21	6	12
10.00	5.00	1.975	1.108	0.77	4.56	94.36	6	18
-5.00	25.00	1.892	0.1185	0.86	26.98	90.44	6	18
0.75	25.00	1.892	0.01777	0.82	27.52	89.49	6	13
10.00	25.00	1.892	0.2369	0.86	26.37	87.33	6	18
-5.00	120.00	1.874	0.02502	0.76	112.28	90.91	6	19
0.75	120.00	1.874	0.003752	0.88	126.99	80.24	6	16
10.00	120.00	1.874	0.05003	0.75	111.72	88.12	6	11

Table 3.4: Frequency domain identification algorithm validation for the FFOPDT plant with $\alpha = 1.00$

K	T	ω_u	A_u	$\tilde{\alpha}$	$\tilde{T}$	%Fit	Points	Steps
-5.00	5.00	1.689	0.5881	0.89	4.56	79.48	5	17
0.75	5.00	1.689	0.08821	0.92	4.43	88.06	6	11
10.00	5.00	1.689	1.176	0.90	4.50	82.65	5	18
-5.00	25.00	1.596	0.1253	0.96	25.71	82.25	6	15
0.75	25.00	1.596	0.01879	0.97	24.52	91.00	6	16
10.00	25.00	1.596	0.2506	0.96	22.87	94.39	6	15
-5.00	120.00	1.576	0.02644	1.03	125.22	93.74	6	13
0.75	120.00	1.576	0.003966	0.94	114.50	83.79	6	13
10.00	120.00	1.576	0.05288	0.94	117.18	80.95	6	13

Table 3.5: Frequency domain identification algorithm validation for the FFOPDT plant with $\alpha = 1.57$

K	T	ω_u	A_u	$\tilde{\alpha}$	$\tilde{T}$	%Fit	Points	Steps
-5.00	5.00	0.8673	1.525	1.03	3.74	-	3	13
0.75	5.00	0.8673	0.2288	0.67	2.92	-	2	19
10.00	5.00	0.8673	3.050	0.83	3.20	-	2	12
-5.00	25.00	0.7196	0.3534	1.68	26.11	56.71	2	15
0.75	25.00	0.7196	0.05301	0.93	15.11	-	3	12
10.00	25.00	0.7196	0.7068	1.44	21.04	69.99	3	15
-5.00	120.00	0.685	0.07637	1.51	113.32	82.40	4	13
0.75	120.00	0.685	0.01145	1.52	118.46	82.86	4	14
10.00	120.00	0.685	0.1527	1.57	115.11	96.48	4	19

In this chapter, some particular issues were identified:

- Example 3.1 clearly shows the importance of choosing a suitable initial model (i.e., initialization) for identification in case of a relatively simple FO system. In this case, forming bounds on the parameter search space can be beneficial, but makes the optimization procedure less efficient. However, the variable transform method may be used in case of unbounded optimization by means of, e.g., the Levenberg-Marquardt algorithm.

- Closed-loop parametric identification depends on characteristics on the actuating device; linear approximations will only usually work in practice in narrow operating regions making the indirect identification approach less useful, see, e.g., Example 6.2 from Chapter 6 for an illustration; several linear approximations are usually required. A more detailed discussion related to control methods may be found later in this work in Chapter 4.

- The presented frequency domain identification method is based on magnitude analysis of the system. Using the proposed relay feedback approach allows identifying frequency response points for frequencies lower than the ultimate one. However, the condition in (3.46) limits the number of points that may be used for identification. The frequencies, where the magnitude response is measured, must be properly spaced. These issues, as well as the effect of the value of the lag L , must be further investigated. Furthermore, the method is to be verified with real-life systems.

Examples including real-life applications and relevant discussion are provided in the Chapters 6 and 7.

Chapter 4

Fractional-order PID Controller Design

In this chapter, the problem of FOPID controller design is investigated. The treated problems include controller design for nonlinear and unstable systems. In addition, a retuning method is proposed, which allows introducing fractional-order dynamics into existing industrial PID control loops thereby increasing control performance without modifying the internal closed-loop control system. Finally, a method for plant analysis and controller design for automatic tuning applications is provided. The chapter has the following structure. The general optimization method based on the Nelder-Mead algorithm is described in Section 4.1. The proposed gain and order scheduling method is given in Section 4.2. An approach for stabilization of unstable plants using the FOPID controller is provided in Section 4.3. The existing control loop retuning method is summarized in Section 4.4. Finally, the methods proposed for FOPID controller automatic tuning applications are given in Section 4.5. Conclusions are drawn in Section 4.6.

4.1 Optimization based Controller Design

There are several aspects to the problem of designing a fractional PID controller by means of optimization:

- The type of plant to be controlled;
- Optimization criteria;
- FOPID design specifications;
- Parameters to optimize;
- Selection of initial parameters in the search space.

A general approach is desired for optimizing performance of a FOPID control loop regardless of the plant type. However, there is a vast array of well-established tuning techniques for common model types [83]. So if a fractional-order model can be approximated by a simple model to a certain degree of validity, it may be used to obtain initial conventional PID controller parameters. These parameters can then be further optimized to achieve better performance.

We consider a general dynamic system of the form

$$\dot{x}(t) = f(x(t), u(t)), \quad y = h(x(t)), \tag{4.1}$$

where $x(t) \in \mathbb{R}^n, u(t) \in \mathbb{R}, f(x, u) \in \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$, and $h : \mathbb{R}^n \to \mathbb{R}$, and a FOPID controller of the form

$$u(t) = K_p e(t) + K_i \mathscr{D}^{-\lambda} e(t) + K_d \mathscr{D}^{\mu} e(t),$$

which corresponds to the parallel form of the FOPID controller in (2.28), and $e(t) = r(t) - y(t)$ is the error signal, and $r(t)$ is the desired control reference. The improvement of control system performance in the time domain is equivalent to the problem of minimizing $e(t)$. Several performance metrics may be considered [6]:

- Integral square error
$$ISE = \int_0^t e^2(t)\mathrm{d}t; \tag{4.2}$$
- Integral absolute error
$$IAE = \int_0^t |e(t)|\,\mathrm{d}t; \tag{4.3}$$
- Integral time-square error
$$ITSE = \int_0^t te(t)^2\mathrm{d}t; \tag{4.4}$$
- Integral time-absolute error
$$ITAE = \int_0^t t\,|e(t)|\,\mathrm{d}t. \tag{4.5}$$

To ensure the stability and robustness of the control system one may introduce frequency domain specifications. Suppose that there exists a FO linear approximation of the system in (4.1) for a working point (u_0, y_0), which describes dynamics of the system reasonably well, and is given by a transfer function $G_p(s)$ of the form (2.13). Then, the following specifications may be imposed based on analysis of the open-loop frequency response $F(j\omega) = C(j\omega)G_p(j\omega)$, where $C(j\omega)$ is the FOPID controller [72]:

- Gain margin A_m, where

$$A_m = 1 - |F(j\omega_g)|\,, \quad \arg(F(j\omega_g)) = -\pi. \tag{4.6}$$

- Phase margin φ_m and gain crossover (critical) frequency ω_c, where

$$\arg(F(j\omega_c)) = -\pi + \varphi_m, \quad |F(j\omega_c)| = 1. \tag{4.7}$$

- Robustness to gain variations of the plant:

$$\left.\frac{\mathrm{d}\arg(F(j\omega))}{\mathrm{d}\omega}\right|_{\omega=\omega_c} = 0. \tag{4.8}$$

- High-frequency noise rejection with noise attenuation of A dB for the complementary sensitivity function

$$\left|T(j\omega) = \frac{F(j\omega)}{1+F(j\omega)}\right|_{\mathrm{dB}} \leqslant A\,\mathrm{dB}, \tag{4.9}$$

  for all frequencies $\omega \geqslant \omega_t$ rad/s.

- Disturbance rejection with a constraint B dB on the sensitivity function

$$\left|S(j\omega) = \frac{1}{1+F(j\omega)}\right|_{\mathrm{dB}} \leqslant B\,\mathrm{dB}, \tag{4.10}$$

  for all frequencies $\omega \leqslant \omega_s$ rad/s.

The optimization problem can thus be formulated as

$$\min_{\theta_c} J_c(\cdot), \tag{4.11}$$

where θ_c is the FOPID parameter set

$$\theta_c = \left[\begin{array}{ccccc} K_p & K_i & K_d & \lambda & \mu \end{array}\right], \tag{4.12}$$

and $J_c(\cdot)$ is a joint cost function that includes the specifications in both time and frequency domains. Additional optimization problems may be formulated by fixing some of the parameters in (4.12), e.g., controller gains.

In addition, in practical problems constraints on the control law $u(t)$ may be imposed in the form

$$u(t) \in [u_{min}; u_{max}]. \tag{4.13}$$

In this work we consider the Nelder-Mead simplex optimization method described in Section 2.5.3. The reasons for using this method are as follows [53]:

- The Nelder-Mead method typically produces significant improvement of a performance measure in industrial control applications within the first few iterations;
- Cost function evaluation is computationally expensive;
- Relatively simple implementation, including embedded device applications.

The conventional design of the method is unbounded in search variables and does not handle constraints. However, following our discussion in Section 2.5.4 bounds are handled by search variable transformation, and constraints are added to the cost by means of penalty functions. In particular, the joint cost function has the form

$$J_c = J_m + J_p, \tag{4.14}$$

where J_m is computed using one of the equations (4.2) through (4.5), and J_p is the nonlinear constraints joint penalty function, formed by a weighted sum of deviations of design specifications in the frequency domain from their prescribed values. Generally, due to the design of the joint cost function J_c, a point in the search space is located, such that satisfies the constraints. Then, the control error $e(t)$ is minimized. This behavior can be controlled by choosing appropriate weights for the performance metric or penalty functions.

Time domain simulation may be carried out using either the Grünwald-Letnikov based solver, described in Section 2.2.3, or using an approximation. In this work we consider only the Oustaloup approximation method in Section 2.3.

One of the known problems of the Nelder-Mead algorithm is the possibility of the following situation [144]:

- The iterates enter a local minimum of the cost;
- Very little progress is made in terms of improving performance over the next iterations.

This makes it difficult to define a general enough stopping criterion for the algorithm. Therefore, a limit on the maximum number of iterations should be introduced.

Consider now a motivating example, where an improvement in performance is achieved by optimizing only the orders of the FOPID controller, whereas the gains correspond to those obtained by means of a conventional tuning formula.

Example 4.1 Consider a fractional-order system of a heating furnace [99, 114, 151]. The normalized model has the form

$$G = \frac{0.59172}{8576.3s^{1.31} + 3555.9s^{0.97} + 1}. \tag{4.15}$$

We shall design a fractional-order controller for this plant using the method described above. First, a conventional FOPDT model approximation is given by

$$G_{FOPDT}(s) = \frac{0.58813}{1 + 4766.83s} e^{-25.1528s}.$$

Using the Ziegler-Nichols PID tuning formula, the integer-order PID parameters are then obtained as $K_p = 386.676, K_i = 50.3056, K_d = 12.5764$. These gains are fixed and orders λ and μ are used as search parameters for the optimization algorithm. The refined Oustaloup filter [72, 146] was used for approximation with $\omega = [0.0001, 10000]$ rad/s, and order $N = 5$. The specification was set for the phase margin, such that $\varphi_m = 75°$. Also, the control signal value was limited to range $u_{lim} = [0; 750]$. A maximum of 100 iterations was considered. The results of controller parameter optimizations using different performance metrics can be found in Table 4.1. Here, N_{iter} denotes the number of iterations until optimization termination, φ_m denotes the phase margin, τ denotes the settling time and σ denotes the overshoot.

Table 4.1: FOPID controller order optimization results for the heating furnace model for different performance metrics

Index	λ	μ	N	φ_m, °	τ, s	ϑ, %
ISE	0.18751	0.43779	36	76.00	1203	12.3
IAE	0.10399	0.45757	50	77.86	1079	10.4
ITSE	0.01000	0.35526	45	79.32	1003	9.1
ITAE	0.11860	0.01071	100	77.36	1118	11.0

The best results were obtained with the ITSE index and the corresponding control system step response and open-loop frequency characteristics can be observed in Figure 4.1 and Figure 4.2, respectively. It is evident that a PD$^\mu$ controller is ultimately obtained. It can be observed that by tuning only the orders of the FOPID controller integrating and differentiating components, a better result was achieved compared to the integer-order tuning.

Further examples are provided in Chapter 7.

4.2 Gain and Order Scheduling

In general, linear controllers will work well with real-life nonlinear systems in a particular working point. If robust control across the full admissible control range is desired, a gain and order scheduling technique may be applied. In the following we provide a summary of the proposed method. Suppose that a nonlinear system is modeled by (4.1). Suppose in addition that a linear

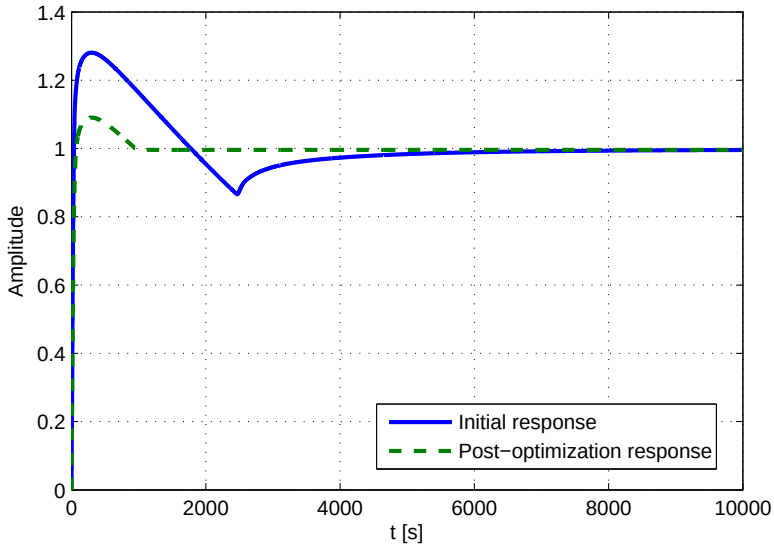


Figure 4.1: Heating furnace control system step response comparison: PID controller with Ziegler-Nichols tuning vs. optimized FOPID controller

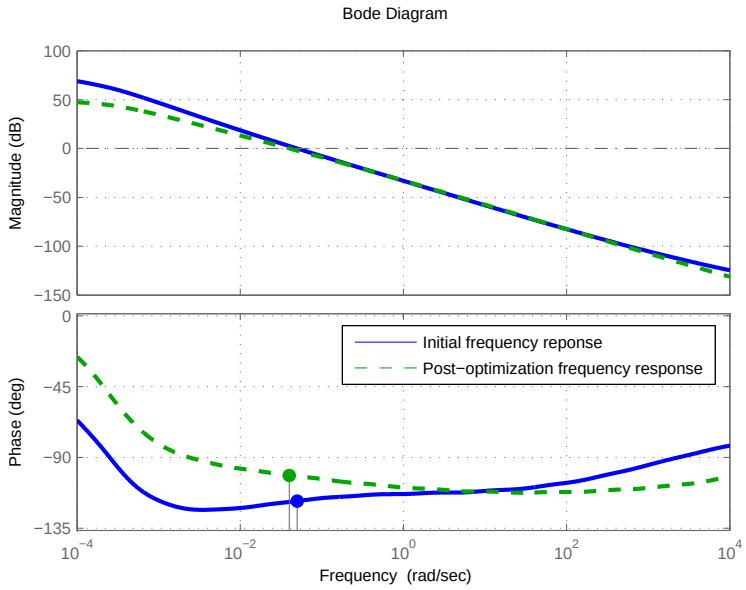


Figure 4.2: Heating furnace control system open-loop frequency domain response comparison: PID controller with Ziegler-Nichols tuning vs. optimized FOPID controller

fractional-order approximations of the form (2.13) may be obtained for a set of working points $\{(u_k; y_k), k = 1, 2, \ldots, n\}$, across the system operating range. Denote by

$$\Psi = \{G_1, G_2, \ldots, G_n\} \tag{4.16}$$

the set of such linear fractional-order approximations. Then, for each $G_i \in \Psi$ design a controller of the form (2.28) that would locally satisfy a set of performance specifications provided in Section 4.1 thereby forming another set denoted by

$$\Omega = \{C_1, C_2, \ldots, C_n\}. \tag{4.17}$$

Now, consider the composite control law

$$\Upsilon(x, s) = \sum_{k=1}^{n} \beta_k(x) C_k(s), \tag{4.18}$$

where $\beta_k(x)$ is a weighting function depending on the scheduled state $x(t)$ and $C_k(s) \in \Omega$.

The choice of n in (4.18) depends on the operating range of the system in (4.1). In the following, we consider the case $n = 2$. Then,

$$\Upsilon(x, s) = \beta_1(x) C_1(s) + \beta_2(x) C_2(s). \tag{4.19}$$

We choose the state $x(t)$ to be the scheduled variable, denote by x_{max} the maximum value, such that $x(t) \in [0, x_{max}]$, and define

$$\beta_1(x) := \frac{(1 - \gamma(x))}{2},\ \beta_2(x) := \frac{\gamma(x)}{2},\ \gamma(x) := \frac{x(t)}{x_{max}}. \tag{4.20}$$

It is obvious that since each entry in (4.17) was designed for a particular linear approximation, the composite control law in (4.18) must be verified across the whole range of linearized models, that is, stability must be ensured for all entries in (4.16). In this work, we consider a heuristic method. Since we employ the negative unity feedback loop, we may compose a set

$$\Lambda = \{\Gamma_1, \Gamma_2, \ldots, \Gamma_{\nu n}\}, \tag{4.21}$$

where

$$\Gamma_k = \frac{Z_k(s)}{P_k(s)} = \frac{\Upsilon_j(x, s) G_k(s)}{1 + \Upsilon_j(x, s) G_k(s)} \tag{4.22}$$

and $j = 1, 2, \ldots, \nu$ is the number of state values considered for the test and Υ_j is a particular control law. For each entry in (4.21) take the characteristic polynomial $P_k(s)$, find the commensurate order $q \geqslant q_{min}$ and use Matignon's theorem from Section 2.2.2.

Remark 4.1 *For the case $n = 2$ in (4.18) using controllers $C_1(s)$ and $C_2(s)$ given by their parameter sets $(K_{p1}, K_{i1}, \lambda_1, K_{d1}, \mu_1)$ and $(K_{p2}, K_{i2}, \lambda_2, K_{d2}, \mu_2)$ we obtain the characteristic polynomial $P(s)$ which depends on the particular parameter set (K, T, α) of the models of the form (2.14) in Ψ with $L = 0$, and on the function $\gamma(x)$ of the scheduled state x. The characteristic polynomial has the following form*

$$P(s) = a_6 s^{\alpha+\lambda_1+\lambda_2} + a_5 s^{\lambda_1+\lambda_2+\mu_2} + a_4 s^{\lambda_1+\lambda_2+\mu_1} + a_3 s^{\lambda_1+\lambda_2} + a_2 s^{\lambda_1} + a_1 s^{\lambda_2}, \quad (4.23)$$

where $a_6 = 2T$, $a_5 = K \cdot K_{d2}\gamma(x)$, $a_4 = K \cdot K_{d1}(1 - \gamma(x))$, $a_3 = 2 + K \cdot K_{p1}(1 - \gamma(x)) + K \cdot K_{p2}\gamma(x)$, $a_2 = K \cdot K_{i2}\gamma(x)$, and $a_1 = K \cdot K_{i1}(1 - \gamma(x))$. The stability test works with commensurate-order systems. When the resulting fractional-order system is not of commensurate order, the stability test produces approximate results [23].

Remark 4.2 *It is important to stress that the controllers in Ω have static parameters and operate simultaneously, while the scheduling, that is the choice of the control action, is done by means of blending functions. Using a static description of the controllers should improve reliability of embedded control implementations* [115]. *Online gain and order scheduling is possible.*

4.3 Stabilization of Unstable Plants

In this section, we propose a method for stabilizing unstable plants $G_u(s)$ of the form (2.13). The task is to make the closed loop system

$$G_{cl}(s) = \frac{C(s)G_u(s)}{1 + C(s)G_u(s)} = \frac{W(s)}{Q(s)} \quad (4.24)$$

stable by a proper choice of parameters of the FOPID controller $C(s)$ of the form (2.28).

To determine stabilizing controllers a randomized method may be used, where FOPID controller parameters are randomly selected from intervals such that $K_p \in [K_p^l, K_p^u]$, $K_i \in [K_i^l, K_i^u]$, $K_d \in [K_d^l, K_d^u]$, $\lambda \in [\lambda^l, \lambda^u]$, $\mu \in [\mu^l, \mu^u]$. Note that the choice of λ and μ should lead to a commensurate-order system, since only then the results of the stability test are reliable. One can also choose a minimum commensurate order $q = 0.01$.

Once a stable point is found, the following procedure is carried out.

1. Two of the controller parameters are parametrized as (p_1, p_2), all other parameters are fixed.

2. A limited number of steps N is selected and a sweep with step sizes Δp_1 and Δp_2 is done from the initial stable point. Four directions are considered. The main idea is illustrated in Figure 4.3:

   (a) Each time only a single parameter is changed.

   (b) If, at any step, an unstable control loop is obtained, then the previous parameter value shall determine the approximate stability boundary for the corresponding direction.

   (c) Otherwise, all points will be tested within the range $\Delta p_1 \cdot N$ and $\Delta p_2 \cdot N$. Testing is done by means of the characteristic polynomial $Q(s)$ of the closed loop system in (4.24) and Matignon's theorem in Section 2.2.2.

3. The stability region will not always have a rectangular shape. To determine the shape every point within the approximate rectangular stability boundary may be tested.

This is a heuristic method similar to [55] and [82].

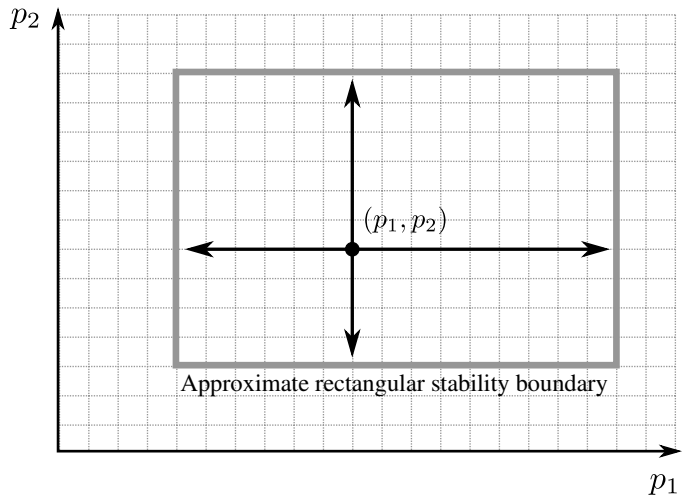


Figure 4.3: Rectangular stability boundaries for stabilization of $G(s)$ using a FOPID controller

Once the procedure is complete, stable parameter ranges are obtained for all controller parameter pairs and may be used in FOPID controller optimization as lower and upper bounds for corresponding controller parameters. Optimizing only two parameters at a time can be beneficial from the perspective of conditioning the problem, albeit in this case it will not be possible to satisfy several design constraints. Optimization may be done using the method described in Section 4.1.

Consider an example, which illustrates the procedure for the determination of stability regions for closed loop control system of an unstable plant.

Example 4.2 Consider an unstable plant given by a fractional-order model

$$G(s) = \frac{s^{0.34} + 0.79}{-2.7s^{1.37} + 0.35s^{0.19} + 1}. \tag{4.25}$$

Using the method above, 10 sets of FOPID controller parameters are randomly chosen such that the parameters fall inside of the following intervals $K_p \in [-10, 10]$, $K_i \in [-10, 10]$, $K_d \in [-10, 10]$, and $\lambda \in [0.2, 1.0]$, $\mu \in [0.5, 1.0]$. A single set is found to produce a stable control system, and the corresponding controller is given by

$$C(s) = -7.17 - 1.56s^{-0.9} - 3.0702s^{0.71}. \tag{4.26}$$

The parameters of this controller form the initial point for the rectangular stability boundary sweep. The sweep is carried out with different parameters for the gain and order step sizes. For the orders, a step of $\Delta p_1^g = \Delta p_2^g = 0.5$ is considered with a maximum number of steps $h_{max}^g = 20$ in each search direction. For the orders, the following parameters are used $\Delta p_1^o = \Delta p_2^o = 0.05$ and $h_{max}^o = 10$. Four pairs of parameter sets are considered: (K_p, K_i), (K_p, K_d), (K_i, K_d), and (λ, μ). The results of the rectangular stability boundary estimation for each set are given in Figures 4.4a through 4.4d. It can be seen that for (K_p, K_i) and (K_p, K_d) the found stability boundary will not be of rectangular shape, which must be taken into account during the controller optimization process. Also note that only a limited number of points is considered, the stability zone may extend beyond the located boundaries.

The located zones may be used for pairwise optimization of FOPID controller parameters. For an illustration, see Chapter 7.

4.4 Retuning FOPID Control for Existing PID Control Loops

Consider a controller represented by $C(s)$ which could either be a classical PI of the form

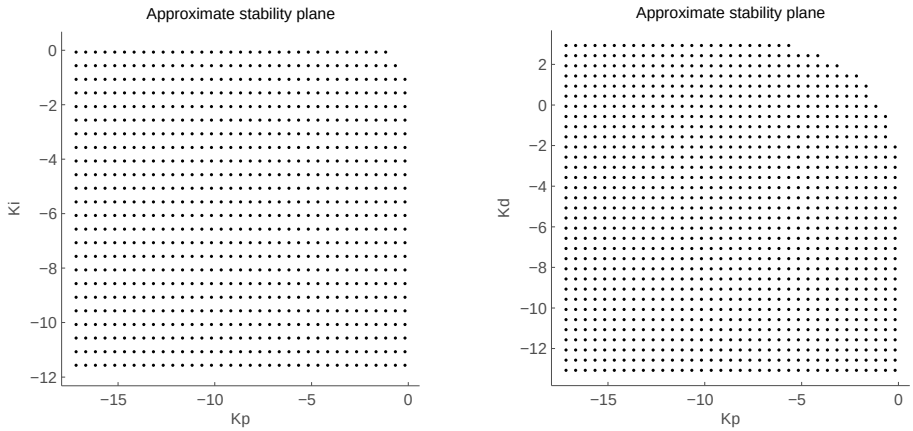
$$C_{PI}(s) = K_p + K_i s^{-1}, \tag{4.27}$$

or PID of the form

$$C_{PID}(s) = K_p + K_i s^{-1} + K_d s, \tag{4.28}$$

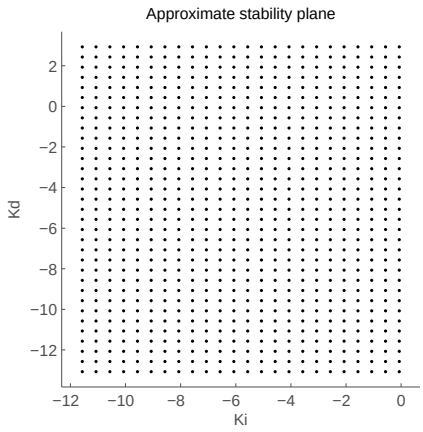
where $K_p, K_i, K_d > 0$ are also assumed within a unity-feedback system

$$G_c(s) = \frac{C(s)P(s)}{1 + C(s)P(s)}. \tag{4.29}$$

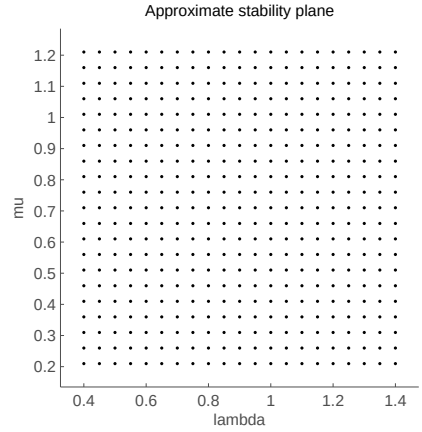


(a) (K_p, K_i)-plane

(b) (K_p, K_d)-plane



(c) (K_i, K_d)-plane



(d) (λ, μ)-plane

Figure 4.4: Determination of rectangular stability boundaries for the stabilizing FOPID controller parameter sets

This feedback control system follows the rules of integer-order differential equations. The objective is to plug in an external fractional-order controller $C_R(s)$ of the form (2.28) into the existing control system in such a way that the dynamics of the overall system follows the rules governed by fractional-order differential equations. The control architecture with an external controller incorporated into an existing feedback control system is shown in Figure 4.5. The method is based on results in [108], where a retuning method for a conventional integer-order PI/PID was studied. The external fractional-order controller $C_R(s)$ captures the input and output signals of the original feedback control system and feeds a corrective signal in addition to the input signal into the feedback control system [29, 40]. The effect of a double feedback configuration in Figure 4.5 is equivalent to a simple unity-gain feedback control system with the controller

$$C^*(s) \quad = \quad (C_R(s) + 1)\, C(s) \tag{4.30}$$

as shown in Figure 4.6.

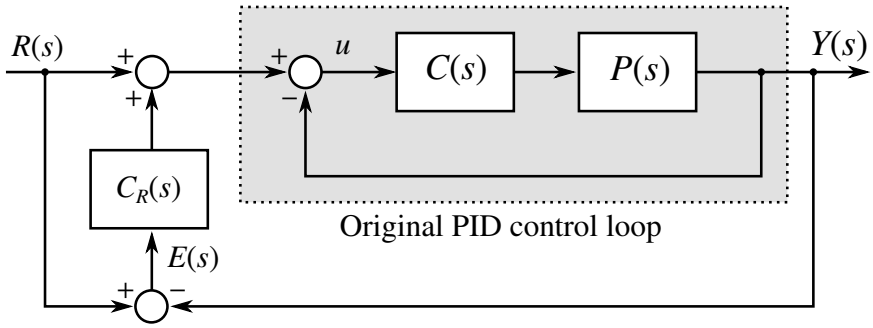


Figure 4.5: The retuning architecture

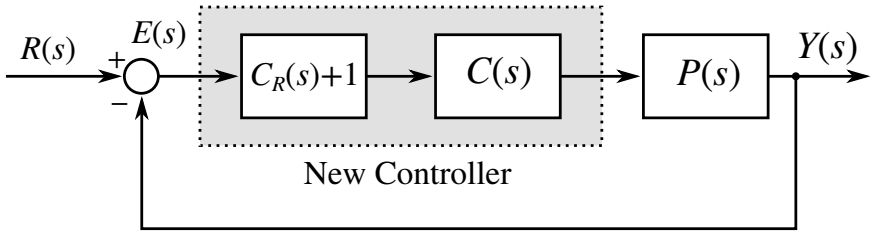


Figure 4.6: Equivalent architecture of the retuning connection

In what follows, several propositions related to the suggested control system retuning architecture are provided.

Proposition 4.1 *(From PI to PI$^\lambda$) Consider the original integer-order PI*

controller in (4.27). *Let* $C_{R1}(s)$ *be a controller of the form*

$$C_{R1}(s) = \frac{K_1 s^{\alpha} + (K_0 - K_p)\, s - K_i}{K_p s + K_i}, \tag{4.31}$$

where the coefficients are $K_p, K_i, K_1, K_0 > 0$ *and the order is* $-1 < \alpha < 1$. *The resulting* PI^{λ} *controller from a classical PI controller with parameters* $K_p, K_i > 0$ *has the following coefficients:*

$$K_p^* = K_0, \quad K_i^* = K_1. \tag{4.32}$$

The order of fractional-order integration is

$$\lambda = 1 - \alpha. \tag{4.33}$$

Proposition 4.2 *(From PI to* $PI^{\lambda}D^{\mu}$*) Consider the original integer-order PI controller in* (4.27). *Let* $C_{R2}(s)$ *be a controller of the form*

$$C_{R2}(s) = \frac{K_2 s^{\beta} + K_1 s^{\alpha} + (K_0 - K_p)\, s - K_i}{K_p s + K_i}, \tag{4.34}$$

where the coefficients are $K_p, K_i, K_2, K_1, K_0 > 0$, *and the orders are* $-1 < \alpha < 1$ *and* $1 < \beta < 2$. *The resulting* $PI^{\lambda}D^{\mu}$ *controller from a classical PI controller with parameters* $K_p, K_i > 0$ *has the following coefficients:*

$$K_p^* = K_0, \quad K_i^* = K_1, \quad K_d^* = K_2. \tag{4.35}$$

The orders of fractional-order integration and differentiation are

$$\lambda = 1 - \alpha, \quad \mu = \beta - 1. \tag{4.36}$$

Proposition 4.3 *(From PID to* $PI^{\lambda}D^{\mu}$*) Consider the original integer-order PID controller in* (4.28). *Let* $C_{R3}(s)$ *be a controller of the form*

$$C_{R3}(s) = \frac{K_2 s^{\beta} + K_1 s^{\alpha} - K_d s^2 + (K_0 - K_p)\, s - K_i}{K_d s^2 + K_p s + K_i}, \tag{4.37}$$

where the coefficients are $K_p, K_i, K_d, K_2, K_1, K_0 > 0$, *and the orders are* $-1 < \alpha < 1$ *and* $1 < \beta < 2$. *The resulting* $PI^{\lambda}D^{\mu}$ *controller from a classical PID controller with parameters* $K_p, K_i, K_d > 0$ *has the following coefficients:*

$$K_p^* = K_0, \quad K_i^* = K_1, \quad K_d^* = K_2. \tag{4.38}$$

The orders of fractional-order integration and differentiation are

$$\lambda = 1 - \alpha, \quad \mu = \beta - 1. \tag{4.39}$$

Proof: In the general case we show that

$$C^*(s) = (C_R(s) + 1)\, C(s) = K_p^* + K_i^* s^{-\lambda} + K_d^* s^{\mu} \tag{4.40}$$

by substituting the expressions for $C_{Rn}(s), n = 1, 2, 3$ into (4.30), from which (4.40) follows. ■

The summary of the retuning algorithm is provided next.

1. Identify the type of fractional-order controller to be used, i.e., PI$^{\lambda}$ or PI$^{\lambda}$D$^{\mu}$, based on the desired control requirements. The choice of whether a PI$^{\lambda}$ or PI$^{\lambda}$D$^{\mu}$ controller shall be used depends on the number and types of criteria to be satisfied and the model of the plant.

2. Solve for the coefficients of the overall controller $C^*(s) = K_p^* + K_i^* s^{-\lambda} + K_d^* s^{\mu}$ using any method based on the plant model and robustness criteria to be satisfied.

3. Calculate the parameters of $C_R(s)$.

An application of the retuning method is provided in Chapter 7.

4.5 Control Loop Analysis and Controller Design in the Frequency Domain for Automatic Tuning Applications in Process Control

In this section two problems are investigated. First, a control system, comprising a FFOPDT-type plant and FOPID controller, is studied, and system frequency domain characteristics are investigated. Second, a method for locating FOPID controller gains corresponding to particular design specifications is proposed. Since algebraic solutions of equations involved in the computations necessary for achieving the above mentioned goals are not available, numerical methods, based on the Newton-Raphson algorithm, are provided. The proposed numerical algorithms are tailored for each particular problem.

4.5.1 Computation of Control System Characteristics

In what follows, the results related to determining the performance criteria of the control system are provided. The types of specifications considered include those discussed in Section 4.1.

Reconsider the FFOPDT model (2.14) in the complex argument $j\omega$ given by

$$G(j\omega) = \frac{K \mathrm{e}^{-Lj\omega}}{T(j\omega)^{\alpha} + 1}, \tag{4.41}$$

where it is assumed that $K > 0$, $T > 0$, $L > 0$, and $\alpha \in (0, 2)$. We begin the analysis by deriving the equations to obtain the magnitude and phase angle of $G(j\omega)$:

$$|G(j\omega)| = \frac{|K|}{\sqrt{1 + T^2\omega^{2\alpha} + 2T\omega^{\alpha}\cos\left(\frac{\alpha\pi}{2}\right)}} \tag{4.42}$$

and

$$\arg\left(G(j\omega)\right) = -L\omega - \tan^{-1}\left(\frac{T\sin\left(\frac{\alpha\pi}{2}\right)}{\omega^{-\alpha} + T\cos\left(\frac{\alpha\pi}{2}\right)}\right). \tag{4.43}$$

In what follows, we derive open-loop characteristics of the plant, based on the magnitude and phase response analysis of (4.41). In essence, this is done by employing (2.22), and isolating the real and complex parts of the resulting expression. The characteristics are then derived from solutions of particular equations. We begin by obtaining the gain crossover frequency, for which it holds $|G(j\omega_c)| = 1$. Solving this equation yields

$$\omega_c = \left(\frac{\sqrt{\Theta(K,\alpha)} - \cos\left(\frac{\alpha\pi}{2}\right)}{T}\right)^{1/\alpha}, \tag{4.44}$$

where $\Theta(K, \alpha) = K^2 + \cos^2\left(\frac{\alpha\pi}{2}\right) - 1$. For $\omega_c \in \mathbb{R}_+$ to exist the following conditions must hold:

$$\Theta(K,\alpha) \geqslant 0, \quad \sqrt{\Theta(K,\alpha)} - \cos\left(\frac{\alpha\pi}{2}\right) \geqslant 0. \tag{4.45}$$

The phase margin φ_m of the system can then be determined from

$$\varphi_m = \pi - \arg\left(G(j\omega_c)\right) + 2\pi n, \quad n \geqslant 0. \tag{4.46}$$

It is more difficult to derive a formula to find the phase crossover frequency, also referred to as the ultimate frequency of the system ω_u, since we need to solve a transcendental equation

$$-L\omega_u - \tan^{-1}\left(\frac{T\sin\left(\frac{\alpha\pi}{2}\right)}{\omega_u^{-\alpha} + T\cos\left(\frac{\alpha\pi}{2}\right)}\right) = -\pi - 2\pi n, \tag{4.47}$$

where n is determined by the requirement to obtain a minimum gain margin $1/|G(j\omega_u)|$ closest to unity. While ω_u is usually obtained during relay autotuning [6], if it is not given, then the following method may be used to compute it from the FFOPDT model parameters.

Consider the problem of finding a root ω^* of a general nonlinear equation $f(\omega) = 0$ under the constraints $\omega > 0$ and $\omega \in (\omega_b, \omega_h)$. To tackle the problem one may employ the Newton-Raphson method [49] which usually provides quadratic convergence to the solution. The process of locating the

root starts at an initial guess ω_0 and is given by the following iterative formula $\omega_{k+1} = \omega_k - f(\omega)\left(f'(\omega)\right)^{-1}$. Once a prescribed iteration limit ν is reached, or the necessary tolerance ϵ is achieved under the condition $f(\omega_k) < \epsilon$, the algorithm shall stop returning the root ω^*. However, there is a drawback of this algorithm such that local minima of $f(\omega)$ may lead to the change of sign of $f'(\omega)$ and a violation of the condition $\omega_m > 0$ may occur at iteration step $m = k + 1$. To rectify this, the locally obtained solution at step n may be replaced such that $\omega_n = \gamma_c \omega_k$, where $\gamma_c \neq 1$ is some predefined positive factor. If as a result of this modification ω_m no longer belongs to the interval (ω_b, ω_h), the process shall fail returning $\omega^* = 0$ thereby indicating that it could not find a solution. The full algorithm is provided in Figure 4.7.

procedure LocateCharacteristic($\omega_0, \gamma_c, \omega_b, \omega_h, f, f'$)
  $\epsilon \leftarrow$ Tolerance, $\nu \leftarrow$ MaxIterations
  $k \leftarrow 0$; $\omega_k \leftarrow \omega_0$
  while $k < \nu$ and $f(\omega) > \epsilon$ **do**
    $\omega_{k+1} \leftarrow \omega_k - f(\omega_k)(f'(\omega_k))^{-1}$
    if $\omega_{k+1} < 0$ **then**
      $\omega_{k+1} \leftarrow \gamma_c \cdot \omega_k$
    end if
    if $\omega_{k+1} < \omega_b$ or $\omega_{k+1} > \omega_h$ **then**
      return 0
    end if
    $k \leftarrow k + 1$
  end while
  return ω_k
end procedure

Figure 4.7: The modified Newton-Raphson method

Next, we introduce a function

$$\upsilon(\omega) = \arg\left(G(j\omega)\right) + \pi + 2\pi n \tag{4.48}$$

and compute its derivative $\mathrm{d}\upsilon(\omega)/\mathrm{d}\omega$. After simplification we arrive at

$$\upsilon'(\omega) = -L - \frac{\alpha T \sin\left(\frac{\alpha\pi}{2}\right)}{\omega\left(2T\cos\left(\frac{\alpha\pi}{2}\right) + \omega^{-\alpha} + T^2\omega^{\alpha}\right)}. \tag{4.49}$$

We may now use the modified Newton's method in Figure 4.7 to find ω_u. Note that to locate the minimum stability margin we need to introduce a modification to the search algorithm, whereby instead of terminating upon obtaining a solution ω_u^* the gain margin $1/\left|G(j\omega)\right|$ at this frequency is checked. If it is found to be less than unity, the iterative process is repeated assigning $\omega_g \leftarrow \omega_u^*$, $\omega_0 \leftarrow \omega_u^*$, and $n \leftarrow n + 1$. This means that the search direction

must be positive. The gain margin and corresponding crossover frequency are then determined by means of

$$K_c = \min\left(\left|1 - 1/G(j\omega_g)\right|, \left|1 - 1/G(j\omega_u)\right|\right). \tag{4.50}$$

The following proposition can be put forth providing the sufficient conditions to ensure convergence of the iterative process in this case.

Proposition 4.4 *A solution ω_u^* of (4.47) exists if the following conditions are satisfied on the interval (ω_b, ω_h)*

$$\arg\left(G(j\omega_b)\right) > -\pi, \ \arg\left(G(j\omega_h)\right) < -\pi. \tag{4.51}$$

Proof: The proof follows from the Intermediate Value Theorem [14]. We have $f_{arg} : \mathbb{R}_+ \to \mathbb{R}$, such that $f_{arg}(\omega) = \arg\left(G(j\omega)\right)$ is continuous with $\omega \in (\omega_b, \omega_h)$. Then, given (4.51), the theorem states that there exists $\omega^* \in (\omega_b, \omega_h)$ such that $f_{arg}(\omega^*) = -\pi$. ■

Remark 4.3 *It is assumed that the interval (ω_b, ω_h) is correctly chosen for modeling and control purposes.*

Once ω_u is computed, the gain margin of the system K_c, also referred to as the ultimate gain, can be obtained as

$$K_c = 1/\left|G(j\omega_u)\right|. \tag{4.52}$$

Thus, we obtain the relative stability criteria. These provide insight into the closed-loop behavior of the plant and, in addition, will serve as decision variables for conventional PI/PID controller design.

Reconsider the controller $C(j\omega)$ in (2.28)

$$C(j\omega) = K_p + K_i(j\omega)^{-\lambda} + K_d\,(j\omega)^{\mu}. \tag{4.53}$$

We proceed to derive the frequency domain characteristics of this controller:

$$|C(j\omega)| = \sqrt{C_R^2(\omega) + C_I^2(\omega)}, \tag{4.54}$$

where

$$C_R(\omega) = K_p + \omega^{-\lambda} K_i \cos\left(\tfrac{\lambda\pi}{2}\right) + \omega^{\mu} K_d \cos\left(\tfrac{\mu\pi}{2}\right) \tag{4.55}$$

and

$$C_I(\omega) = -\omega^{-\lambda} K_i \sin\left(\tfrac{\lambda\pi}{2}\right) + \omega^{\mu} K_d \sin\left(\tfrac{\mu\pi}{2}\right) \tag{4.56}$$

and for the phase angle as

$$\arg\left(C(j\omega)\right) = \tan^{-1}\left(\frac{C_N(\omega)}{C_D(\omega)}\right), \tag{4.57}$$

where

$$C_N(\omega) = \omega^{\lambda+\mu} K_d \sin\left(\tfrac{\mu\pi}{2}\right) - K_i \sin\left(\tfrac{\lambda\pi}{2}\right) \tag{4.58}$$

and

$$C_D(\omega) = K_i \cos\left(\tfrac{\lambda\pi}{2}\right) + \omega^{\lambda}\left(\omega^{\mu} K_d \cos\left(\tfrac{\mu\pi}{2}\right) + K_p\right). \tag{4.59}$$

We now derive the equations to compute the critical frequencies and corresponding stability margins of the open-loop control system given by $G_{ol}(j\omega) = C(j\omega)G(j\omega)$. A function $\psi_{pm}(\omega)$ for the phase margin is defined as

$$\psi_{pm}(\omega) := |C(j\omega)| \cdot |G(j\omega)| - 1 \tag{4.60}$$

and the derivative thereof is computed yielding

$$\psi'_{pm}(\omega) = A'_1(\omega)\,|G(j\omega)| + |C(j\omega)|\,A'_2(\omega), \tag{4.61}$$

where

$$A'_1(\omega) = A_{11}(\omega)/\,|C(j\omega)| \tag{4.62}$$

with

$$\begin{aligned} A_{11}(\omega) = \omega^{-1-2\lambda}\Bigg(&\mu\omega^{2(\lambda+\mu)}K_d^2 - \lambda K_i\left(K_i + \omega^{\lambda}K_p\cos\left(\tfrac{\lambda\pi}{2}\right)\right) \\ &+ \omega^{\lambda+\mu}K_d\Bigg((\lambda-\mu)\,K_i\sin\left(\tfrac{(\lambda+\mu-1)\pi}{2}\right) \\ &+ \mu\omega^{\lambda}K_p\cos\left(\tfrac{\mu\pi}{2}\right)\Bigg)\Bigg) \end{aligned} \tag{4.63}$$

and

$$A'_2(\omega) = -\frac{\left(T\alpha\omega^{\alpha-1}\left(T\omega^{\alpha} + \cos\left(\tfrac{\alpha\pi}{2}\right)\right)\right)|G(j\omega)|}{P(\omega)}, \tag{4.64}$$

where

$$P(\omega) = 1 + T^2\omega^{2\alpha} + 2T\omega^{\alpha}\cos\left(\tfrac{\alpha\pi}{2}\right). \tag{4.65}$$

We can now use the modified Newton method in Figure 4.7 to solve the equation $\psi_{pm}(\omega_c) = 0$. We suggest to use $\omega_0 = \omega_b$ as the initial estimate, and a factor $\gamma_c > 1$.

Proposition 4.5 *A solution ω_c^* of $\psi_{pm}(\omega_c) = 0$ exists if the following conditions hold on the interval $\omega \in (\omega_b, \omega_h)$*

$$|C(j\omega_b)| \cdot |G(j\omega_b)| > 1 \tag{4.66}$$

and

$$|C(j\omega_h)| \cdot |G(j\omega_h)| < 1. \tag{4.67}$$

Proof is done along the same lines as for Proposition 4.4.
When ω_c^* is obtained, the phase margin is determined in the usual way.

Remark 4.4 *The failure of conditions in* (4.66) *and* (4.67) *or lack of solution* ω_c^* *may indicate unacceptable performance or instability of the closed-loop system.*

Next, we define a function for the gain margin as

$$\psi_{gm}(\omega) = \arg\left(C(j\omega)\right) + \arg\left(G(j\omega)\right) + \pi + 2\pi n. \tag{4.68}$$

Its derivative is given by

$$\psi'_{gm}(\omega) = B'_1(\omega) + \upsilon'(\omega), \tag{4.69}$$

where $B_1(\omega) = \arg\left(C(j\omega)\right)$ and the expression for computing $\upsilon'(\omega)$ has already been provided in (4.49). Therefore, we only need to evaluate $B'_1(\omega)$. The following result is obtained after simplification:

$$B'_1(\omega) = B_{10}/B_{20}, \tag{4.70}$$

where

$$B_{10} = \lambda K_p K_i \sin\left(\tfrac{\lambda\pi}{2}\right) + \omega^{\mu} K_d B_{11} \tag{4.71}$$

with

$$B_{11} = (\lambda+\mu) K_i \cos\left(\tfrac{(\lambda+\mu-1)\pi}{2}\right) + \mu\omega^{\lambda} K_p \sin\left(\tfrac{\mu\pi}{2}\right) \tag{4.72}$$

and

$$\begin{aligned} B_{20} \;=\; & \omega\Bigg(\omega^{\lambda+2\mu}K_d^2 + \omega^{-\lambda}K_i^2 + 2K_pK_i\cos\left(\tfrac{\lambda\pi}{2}\right) + \omega^{\lambda}K_p^2 \\ & -2\omega^{\mu}K_d\Bigg(K_i\sin\left(\tfrac{(\lambda+\mu-1)\pi}{2}\right) - \omega^{\lambda}K_p\cos\left(\tfrac{\mu\pi}{2}\right)\Bigg)\Bigg). \end{aligned} \tag{4.73}$$

We may now use the modified Newton method to solve the equation $\psi_{gm}(\omega_u) = 0$. If one wishes to establish the minimum gain margin according to the earlier discussion, then it is necessary to choose $\omega_0 = \omega_b$ as the initial estimate, and a factor $\gamma_c > 1$.

Proposition 4.6 *A solution* ω_u^* *of* $\psi_{gm}(\omega_u) = 0$ *exists if the following conditions hold on the interval* (ω_b, ω_h)

$$\arg\left(C(j\omega_b)\right) + \arg\left(G(j\omega_b)\right) > -\pi \tag{4.74}$$

and

$$\arg\left(C(j\omega_h)\right) + \arg\left(G(j\omega_h)\right) < -\pi. \tag{4.75}$$

Proof is done along the same lines as for Proposition 4.4.

Remark 4.5 *If at ω_h the argument of $C(j\omega)G(j\omega)$ is greater than $-\pi$, then gain margin may be sufficiently large. This may happen when the time delay L of the system in* (4.41) *is negligible.*

Remark 4.6 *Failure of condition in* (4.67) *to hold indicates that the closed-loop system is most likely unstable, therefore computation of gain margin may not hold any merit.*

The robustness to gain variations specification is given as

$$\left(\frac{\mathrm{d}\arg C(j\omega)G(j\omega)}{\mathrm{d}\omega}\right)_{\omega=\omega_c} = 0. \tag{4.76}$$

Note that the derivative in (4.76) may be computed using the expression in (4.69) taking $\omega = \omega_c$, so no further derivations are required.

Finally, constraints on the sensitivity functions allow to account for output disturbance rejection and high-frequency noise rejection of the control system and are important robustness criteria. Therefore, deriving the equations to compute the magnitudes of these functions at particular frequencies is important. We first consider the sensitivity function $S(j\omega)$ which is given by

$$S(j\omega) = \frac{1}{Q(j\omega)}, \tag{4.77}$$

where $Q(j\omega) = 1 + C(j\omega)G(j\omega)$ is the characteristic polynomial. We proceed as follows

$$|Q(j\omega)| = \frac{\sqrt{Q_R^2(\omega) + Q_I^2(\omega)}}{\sqrt{P(\omega)}}, \tag{4.78}$$

where

$$\begin{aligned} Q_R(\omega) = \cos\left(\tfrac{\lambda\pi}{2}\right) - T\omega^{\alpha}\sin\left(\tfrac{(\lambda+\alpha-1)\pi}{2}\right) \\ + \omega^{\lambda}KK_d\cos\left(L\omega - \tfrac{(\lambda+\mu)\pi}{2}\right) \\ + KK_i\omega^{-\lambda}\cos(L\omega) + KK_p\cos\left(L\omega - \tfrac{\lambda\pi}{2}\right) \end{aligned} \tag{4.79}$$

and

$$\begin{aligned} Q_I(\omega) = T\omega^{\alpha}\cos\left(\tfrac{(\lambda+\alpha-1)\pi}{2}\right) + \sin\left(\tfrac{\lambda\pi}{2}\right) \\ - KK_d\omega^{\mu}\sin\left(L\omega - \tfrac{(\lambda+\mu)\pi}{2}\right) \\ - KK_i\omega^{-\lambda}\sin(L\omega) - KK_p\sin\left(L\omega - \tfrac{\lambda\pi}{2}\right) \end{aligned} \tag{4.80}$$

and $P(\omega)$ is obtained by means of (4.65). Therefore, the magnitude of the sensitivity function at a particular frequency ω may be determined from

$$|S(j\omega)| = \frac{1}{|Q(j\omega)|}. \tag{4.81}$$

For the complementary sensitivity function $T(j\omega)$ it holds that

$$|T(j\omega)| = \frac{|C(j\omega)|\,|G(j\omega)|}{|Q(j\omega)|}. \tag{4.82}$$

By using (4.81) and (4.82) we may evaluate the magnitude response in the range of interest, usually in the interval given by (ω_b, ω_h) and further constrain the optimization problem by including disturbance and noise rejection specifications.

In the following, a tuning method based on the above computations is proposed. Our reasoning here is along the lines of [111]. Here is the summary of the method.

1. Consider the parameters ω_u and K_c of the FFOPDT model in (2.14). If they are not known, compute them using the method described above;

2. Compute the parameter set $\theta_G = \{K_p, K_i, K_d\}$ of a conventional PID-type controller;

3. Design a cost function $J(\cdot)$ based on desired system performance according to frequency domain specifications;

4. Do a sweep of parameters in the set $\theta_P = \{\lambda, \mu\}$ within a predefined region, computing the chosen design specifications;

5. Choose the best controller according to $\min J(\cdot)$.

This method is justified because in case of a model (2.14) we have a fractional plant to control. Therefore, optimizing the orders of the controller may lead to the improvement of performance and robustness of the control system as it was shown in Example 4.1. The method is much less computationally intensive than general nonlinear optimization. However, it also essentially complements said optimization process by shifting the initial points (λ, μ) the parameter search space and thereby moving away from a potential local minimum of the cost function [126]. Therefore, the fractional power sweep procedure may be used as the first step in the optimization sequence. In such a case it is sufficient to choose a relatively large sweep step size, e.g., $\Delta\gamma = 0.1$.

4.5.2 FOPID Controller Design

The parameters of the FFOPDT plant in (4.41) are assumed to be obtained by means of a relay autotuning algorithm considered in Section 3.4. As we have seen, by properly identifying several points (ω_k, A_k) in the frequency domain it is possible to determine not only the classical FOPDT model, but also the fractional order α of the FFOPDT model. In case of the conventional model, parameters, such as plant gain K_c and lag L_c may be determined experimentally and will coincide with corresponding parameters of the FFOPDT model. The time constant T_c may be computed, when the frequency ω_u, corresponding to the ultimate gain of the system, is found, by means of

$$T_c = \frac{\tan(\pi - L_c\omega_u)}{\omega_u}. \tag{4.83}$$

Once all of the conventional FOPDT model parameters are obtained, we can use the F-MIGO rule proposed in [72] for FOPI controllers to determine the appropriate integrator order λ of the controller in (2.28) by considering basic plant dynamics through the relative dead time parameter τ_c:

$$\tau_c = \frac{L_c}{L_c + T_c}. \tag{4.84}$$

The following approximate rule is then employed:

$$\lambda = \begin{cases} 1.1, & \tau_c \geqslant 0.6, \\ 1.0, & 0.4 \leqslant \tau_c < 0.6, \\ 0.9, & 0.1 \leqslant \tau_c < 0.4, \\ 0.7, & \tau_c < 0.1. \end{cases} \tag{4.85}$$

This holds under the assumption that conventional plant dynamics are described by T_c with reasonable accuracy. Regardless of the situation, this is an approximation, therefore one may apply this rule to find an initial value of λ, which may be additionally tuned.

For deciding the differentiator order μ knowledge of the FFOPDT plant order α is necessary. The following relation should hold:

$$\mu \leqslant \alpha. \tag{4.86}$$

The particular value of μ may be chosen according to the design specifications imposed on the control system. In what follows, we consider the following particular specifications, discussed in more detail in Section 4.1:

- Exact phase margin φ_m and corresponding crossover frequency ω_c;

- Robustness to gain variations, that is there exists a requirement such that

$$\psi_g'(\omega_c) = 0, \tag{4.87}$$

where

$$\psi_g(\omega) = \arg\left(C(j\omega)\right) + \arg\left(G(j\omega)\right) + \pi + 2\pi n, \tag{4.88}$$

the equations to compute (4.87) are provided in [123].

- Minimal gain margin G_m.

Based on these specifications, the following functions may be defined:

$$\kappa_1(K_p, K_i, K_d) = |C(j\omega)| \cdot |G(j\omega)| - 1, \tag{4.89}$$
$$\kappa_2(K_p, K_i, K_d) = \arg\left(C(j\omega)\right) + \arg\left(G(j\omega)\right) + \pi - \varphi_m - 2\pi n, \tag{4.90}$$
$$\kappa_3(K_p, K_i, K_d) = \psi_{gm}'(\omega), \tag{4.91}$$

where $\omega = \omega_c$. Note that this does not include the gain margin specification. However, the solution is only considered feasible, if the minimal gain margin is satisfied.

To find the gains $g = \begin{bmatrix} K_p & K_i & K_d \end{bmatrix}^{\mathrm{T}}$ of the FOPID controller according to the specifications given above it is necessary to solve a system of nonlinear equations comprised of the design specification functions

$$F_s = \begin{bmatrix} \kappa_1(\cdot) & \kappa_2(\cdot) & \kappa_3(\cdot) \end{bmatrix}^{\mathrm{T}} = 0. \tag{4.92}$$

To this end, Newton's method in several dimensions may be employed. Beginning from the initial estimate g_0 the iterative process begins, until a particular stop condition is satisfied. On every step, a linear system

$$J\Delta g = -F_s \tag{4.93}$$

must be solved. Then the new controller gain vector g^+ is computed as

$$g^+ = g + \Delta g. \tag{4.94}$$

In the following, we provide all of the elements of the Jacobian matrix J comprised of the partial derivatives such that $J_{n,1} = \partial\kappa_n/\partial K_p$, $J_{n,2} = \partial\kappa_n/\partial K_i$, and $J_{n,3} = \partial\kappa_n/\partial K_d$, for $n = 1, 2, 3$:

$$J_{1,1} = \frac{A_G A_{CR}}{A_C},\ J_{1,2} = \frac{A_G A_{12}}{A_C},\ J_{1,3} = \frac{A_G A_{13}}{A_C}, \tag{4.95}$$

where $A_G = |G(j\omega)|$ in (4.42), $A_{CR} = C_R(\omega)$ in (4.55), and $A_C = |C(j\omega)|$ in (4.54),

$$A_{12} = \omega^{-2\lambda}\Big(-\omega^{\lambda+\mu}\sin\Big(\tfrac{(\lambda+\mu-1)\pi}{2}\Big)K_d + K_i + \omega^{\lambda}\cos\big(\tfrac{\lambda\pi}{2}\big)K_p\Big), \tag{4.96}$$

$$A_{13} = \omega^{-\lambda+\mu}\Big(\omega^{\lambda+\mu}K_d - \sin\Big(\tfrac{(\lambda+\mu-1)\pi}{2}\Big)\,K_i + \omega^{\lambda}\cos\big(\tfrac{\mu\pi}{2}\big)\,K_p\Big). \tag{4.97}$$

Then,

$$J_{2,1} = \frac{A_{21}}{A_2},\ J_{2,2} = -\frac{A_{22}}{A_2},\ J_{2,3} = \frac{A_{23}}{A_2}, \tag{4.98}$$

where

$$A_{21} = \omega^{\lambda}\left(-\omega^{\lambda+\mu}\sin\big(\tfrac{\mu\pi}{2}\big)\,K_d + \sin\big(\tfrac{\lambda\pi}{2}\big)\,K_i\right), \tag{4.99}$$

$$A_{22} = \omega^{\lambda}\Big(\omega^{\mu}\cos\Big(\tfrac{(\lambda+\mu-1)\pi}{2}\Big)\,K_d + \sin\big(\tfrac{\lambda\pi}{2}\big)\,K_p\Big), \tag{4.100}$$

$$A_{23} = \omega^{\lambda+\mu}\Big(\cos\Big(\tfrac{(\lambda+\mu-1)\pi}{2}\Big)\,K_i + \omega^{\lambda}\sin\big(\tfrac{\mu\pi}{2}\big)\,K_p\Big), \tag{4.101}$$

and

$$\begin{aligned} A_2 = {} & \omega^{2(\lambda+\mu)}K_d^2 + K_i^2 + 2\omega^{\lambda}\cos\big(\tfrac{\lambda\pi}{2}\big)\,K_iK_p + \omega^{2\lambda}K_p^2 \\ & + 2\omega^{\lambda+\mu}K_d\Big(-\sin\Big(\tfrac{(\lambda+\mu-1)\pi}{2}\Big)\,K_i + \omega^{\lambda}\cos\big(\tfrac{\mu\pi}{2}\big)\,K_p\Big). \end{aligned} \tag{4.102}$$

Finally,

$$J_{3,1} = \frac{A_{31}}{A_3},\ J_{3,2} = \frac{A_{32}}{A_3},\ J_{3,3} = \frac{A_{33}}{A_3}, \tag{4.103}$$

where

$$\begin{aligned} A_{31} = {} & \omega^{\lambda-1}\Big(\mu\omega^{3(\lambda+\mu)}\sin\big(\tfrac{\mu\pi}{2}\big)\,K_d^3 - \omega^{2(\lambda+\mu)}\Big(2\mu\sin\big(\tfrac{\lambda\pi}{2}\big) \\ & + \lambda\sin\Big(\tfrac{(\lambda+2\mu)\pi}{2}\Big)\Big)K_d^2K_i + \lambda\sin\big(\tfrac{\lambda\pi}{2}\big)\,K_i\Big(K_i^2 - \omega^{2\lambda}K_p^2\Big) \\ & - \omega^{\lambda+\mu}K_d\Big(\Big(2\lambda\sin\big(\tfrac{\mu\pi}{2}\big) + \mu\sin\Big(\tfrac{(2\lambda+\mu)\pi}{2}\Big)\Big)K_i^2 \\ & + 2(\lambda+\mu)\omega^{\lambda}\cos\Big(\tfrac{(\lambda+\mu-1)\pi}{2}\Big)\,K_iK_p + \mu\omega^{2\lambda}\sin\big(\tfrac{\mu\pi}{2}\big)\,K_p^2\Big)\Big), \end{aligned} \tag{4.104}$$

$$\begin{aligned} A_{32} = {} & \omega^{\lambda-1}\Big((\lambda+\mu)\omega^{2\lambda+3\mu}\cos\Big(\tfrac{(\lambda+\mu-1)\pi}{2}\Big)\,K_d^3 \\ & + \omega^{2(\lambda+\mu)}\Big(2(\lambda+\mu)\sin\big(\tfrac{\lambda\pi}{2}\big) + \lambda\sin\Big(\tfrac{(\lambda+2\mu)\pi}{2}\Big)\Big)K_d^2K_p \\ & + \lambda\sin\big(\tfrac{\lambda\pi}{2}\big)\,K_p(-K_i^2 + \omega^{2\lambda}K_p^2) \\ & + \omega^{\mu}K_d\Big(-(\lambda+\mu)\cos\Big(\tfrac{(\lambda+\mu-1)\pi}{2}\Big)\,K_i^2 - 2\mu\omega^{\lambda}\sin\big(\tfrac{\mu\pi}{2}\big)\,K_iK_p \\ & + \omega^{2\lambda}\Big(2\lambda\cos\Big(\tfrac{(\lambda+\mu-1)\pi}{2}\Big) + (\lambda+\mu)\sin\Big(\tfrac{(\lambda-\mu)\pi}{2}\Big)\Big)K_p^2\Big)\Big), \end{aligned} \tag{4.105}$$

$$A_{33} = \omega^{\lambda+\mu-1}\Big((\lambda+\mu)\cos\Big(\tfrac{(\lambda+\mu-1)\pi}{2}\Big)K_i^3 - 2\lambda\omega^{2\lambda+\mu}\sin\big(\tfrac{\lambda\pi}{2}\big)K_pK_iK_d + \omega^{\lambda}\Big(2(\lambda+\mu)\sin\big(\tfrac{\mu\pi}{2}\big) + \mu\sin\Big(\tfrac{(2\lambda+\mu)\pi}{2}\Big)\Big)K_i^2K_p + \omega^{2\lambda}\Big(2\mu\cos\Big(\tfrac{(\lambda+\mu-1)\pi}{2}\Big) - (\lambda+\mu)\sin\Big(\tfrac{(\lambda-\mu)\pi}{2}\Big)\Big)K_iK_p^2 + \mu\omega^{3\lambda}\sin\big(\tfrac{\mu\pi}{2}\big)K_p^3 - \omega^{2(\lambda+\mu)}K_d^2\Big((\lambda+\mu)\cos\Big(\tfrac{(\lambda+\mu-1)\pi}{2}\Big)K_i + \mu\omega^{\lambda}\sin\big(\tfrac{\mu\pi}{2}\big)K_p\Big)\Big), \quad (4.106)$$

and

$$A_3 = A_2^2. \quad (4.107)$$

We can define a stopping criterion for the iterative process as a condition on the square norm of (4.92):

$$\|F_s(\cdot)\|_2 < \epsilon. \quad (4.108)$$

There is no feasible way to determine whether a solution to (4.92) exists. However, by virtue of the Inverse Value Theorem [76] it is possible to claim that if the Jacobian does not vanish at g_0, a local minimum of $\|F_s(\cdot)\|_2$ will be found around g_0. It is up to the user to check, whether the obtained set of controller parameters is feasible or not. If no feasible solution is obtained during optimization, a new initial estimate should be selected.

The complete optimization algorithm is presented in Figure 4.8. The meaning of procedure return codes is provided in Table 4.2.

Table 4.2: Meaning of optimization procedure return codes

Code	Description
-2	Additional condition not satisfied—the gain margin G_m^* computed for the control system is less than the value given in G_m.
-1	Singular Jacobian matrix—local minimum possible.
0	Maximum number of algorithm iterations reached.
1	All conditions satisfied, successful termination.

Consider now two examples, which illustrate the use of the methods discussed above.

procedure FOPIDDesign($g_0, \omega_c, \varphi_m, G_m$)
  $\epsilon \leftarrow$ Tolerance, $\epsilon_m \leftarrow$ MachineTolerance
  $g \leftarrow g_0, k \leftarrow 0, \nu \leftarrow$ MaxIterations
  while $k < \nu$ **do**
    if $\det J < \epsilon_m$ **then return** $\{-1, g\}$
    end if
    if $G_m^* < G_m$ **then return** $\{-2, g\}$
    end if
    if $\|F_s\|_2 < \epsilon$ **then return** $\{1, g\}$
    end if
    $g \leftarrow g - J^{-1}F_s$
    $k \leftarrow k + 1$
  end while
  return $\{0, g\}$
end procedure

Figure 4.8: Determination of FOPID controller gains

Example 4.3 We consider here a FFOPDT model of a heating process from [61]. It is given by the following FO transfer function:

$$G(s) = \frac{66.16}{12.72s^{0.5} + 1} \mathrm{e}^{-1.93s}. \tag{4.109}$$

We use the tuning method from Section 4.5.1, choosing three different sets of conventional PI tuning rules. Namely, the classical Ziegler-Nichols method [152], the Cohen-Coon method [146] and the AMIGO method [6]. To compute the controller gains we need to recover the ultimate frequency ω_u and gain margin K_c of the model in (4.109). The weighted cost function is given by

$$J = w_1 \left|\varphi_m - \tilde{\varphi}_m\right| + w_2 \left|\left(\mathrm{d}\arg\left(C(j\omega)G(j\omega)\right)/\mathrm{d}\omega\right)_{\omega=\omega_c}\right|,$$

where $w_1 = 100/\pi$, $w_2 = 10$. The range of the integrator order sweep is selected as $\lambda \in [0.5, 1.5]$ with a step size of $\Delta\lambda = 0.05$. The desired phase margin is $\varphi_m = 75°$. In addition, any λ that yields a control system with a gain margin such that $G_m < 2.5$ will be discarded. The results of controller design are presented in Table 4.3.

As it can be seen, it was not possible to stabilize the control system in case of the Ziegler-Nichols tuning rules. However, in case of the AMIGO and Cohen-Coon tuning an suboptimal λ subject to given frequency domain specifications, that is, the phase margin specification, was found.

Example 4.4 Reconsider the model in (4.109). In this example, the design specifications are chosen as $\omega_c = 0.1$, $\varphi_m = 60°$, $G_m \geqslant 10$dB, and the robustness to gain variations criterion must also be fulfilled.

Table 4.3: Controller design by testing fractional integrator powers

Method	K_p	K_i	λ^*	φ_m	G_m
Ziegler-Nichols	0.4059	2.2021	Cannot stabilize		
AMIGO	0.0024	0.0036	0.80	73.9°	30.7 dB
Cohen-Coon	0.0129	0.0375	0.55	74.7°	12.0 dB

Suppose that an autotuning procedure is employed. For the plant (4.109) we found the ultimate frequency ω_u to be approximately equal to 7.85rad/s. Using (4.83) the time constant of the conventional FOPDT plant would be computed as $T_c = 0.0794$s, which obviously provides the wrong description of the plant dynamics. Therefore, using the F-MIGO rule does not hold any merit. Using a more sophisticated approach for computing T_c involving identification of several points on the Nyquist curve, however, yields a value $\tilde{T}_c \approx 12$, therefore we have $\lambda = 0.9$. The differentiator order is then chosen as $\mu = 0.5$. The initial gains for optimization are selected such that $g_0 = \begin{bmatrix} 1/K & 1/K & 1/K \end{bmatrix}$. The optimization procedure is then employed. In 4 iterations the norm condition $\epsilon = 10^{-4}$ is satisfied, and the gains of the resulting controller are obtained:

$$K_p = -0.002934,\ K_i = 0.01030,\ K_d = 0.05335. \tag{4.110}$$

The Bode diagram depicting the open-loop frequency response of the control system is given in Figure 4.9. It may be seen that the specifications are fulfilled, with $G_m = 11.5 > 10$dB.

These examples are further extended in Chapter 5, namely by Examples 5.6 and 5.7, where the results of time domain simulations of the designed control systems are provided.

4.6 Conclusions

In this chapter, several methods for FOPID controller design were proposed. In particular, a general Nelder-Mead simplex optimization algorithm based method was described, which takes into account specifications in both time and frequency domains. This is a novel FOPID controller optimization approach that does not require computation of the gradient, and is therefore capable of handling nonsmooth problems, which may be encountered when tuning FOPID controllers for nonlinear systems. Several other methods that are frequently used to solve industrial control problems were also studied in the context of FOPID control: gain and order scheduling, stabilization, and control loop retuning—all of which leverage the proposed optimization

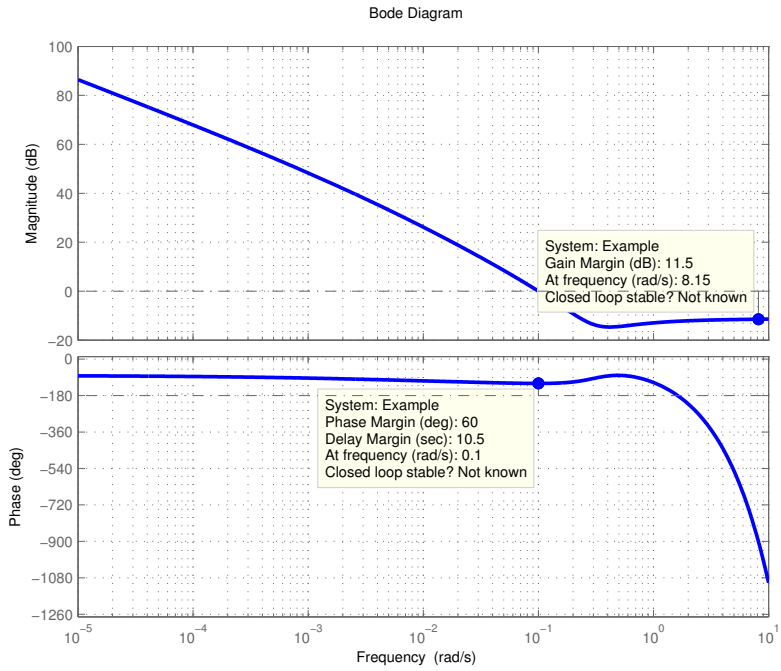


Figure 4.9: Controller design results: Bode diagram of the open-loop control system

algorithm. The combination of these methods leads to a unified approach for designing FO controllers for industrial control applications. This approach is similar to [90], but the important difference is that the methods revolve around a FO extension to conventional PID control, thus potentially facilitating industrial integration of the results of this work.

The use of Nelder-Mead direct search method is well justified by the fact that time domain evaluation of FO control systems is relatively expensive. The method is also easy to implement on embedded hardware. It appears to be quite robust in handling the FOPID controller design problem. A more important issue is the selection of appropriate design specifications. This leads to the problem of determining the existence of a feasible and (sub)optimal solution for a control design task subject to a particular set of specifications. Results from, e.g., [56] may be adopted to tackle this issue.

It was shown that it is possible to combine conventional PID tuning rules with the flexibility of FOPID controllers to achieve performance improvement of the control loop for a plant which has fractional dynamics as seen from Example 4.1. Generally, FOPID controllers offer more tuning flexibility for conventional, integer-order plants as well. However, in order to leverage these favorable qualities high-order models of these plants should be considered,

i.e., models, that take into account more complex dynamics. In addition, one may consider applying FO control to stabilization problems, since FO systems have a larger stability zone. This is illustrated in Example 4.2.

Several other issues were identified and include the following:

- A more general stability analysis scheme is required for the proposed gain and order scheduled approach; this is also true for the problem of determining the stability zone in case of the random stabilizing FOPID controller point search method;
- The retuning method is based on linear system analysis, which may lead to problems with various nonlinear phenomena appearing in real-life nonlinear systems;

In addition, several tuning methods for plants described by FFOPDT models were proposed, including a general method for evaluating the design specifications of such control systems, and a Newton method based approach for locating the gains of the FOPID controller subject to particular design specifications. The latter may be used as a part of a more general tuning algorithm, since it works very well in the vicinity of a solution. However, the tuning algorithm from Section 4.5 needs to be updated to include proper selection of the controller orders λ and μ, which currently relies on approximations.

The mentioned issues require further investigation. Practical applications of the proposed methods are presented in Chapter 7.

Chapter 5

Implementation of Fractional-order Models and Controllers

In this chapter, methods for hardware implementation of fractional-order models are discussed. Both analog and digital realizations are considered. The structure of the chapter is as follows. First, the updated Carlson's method for first-order implicit FO transfer function approximation by means of a modified Newton's method is discussed in Section 5.1. The framework for efficient analog implementation of fractional-order systems is presented in Section 5.2. The proposed digital controller implementation method is described in Section 5.3. The hardware platform used for real-time control experiments is presented in Section 5.4. The development of a FOPID controller prototype based on this method is discussed in Section 5.5. Finally, in Section 5.6 conclusions are drawn.

5.1 An Update to Carlson's Approximation Method for Analog Implementations

In this section, we propose a method, which may be used to obtain analog approximations of fractional-order operators in a similar way to Oustaloup's method in Section 2.3. The developed method is particularly useful in approximation of FO lead-lag compensators.

The initial fractional capacitor approximation algorithm developed by Carlson in [19] relies on a a modified Newton method is proposed in [127, 142]. The method offers convergence of the sequence $\{x_k\}$ that is more rapid than that resulting from using Newton's update formula in (2.36).

The corresponding formula is called Halley's formula:

$$x^+ = x - \frac{f(x)}{f'(x) - \frac{f(x)f''(x)}{2f'(x)}}. \tag{5.1}$$

Consider now a problem of finding an nth root of a real number. The corresponding function is $f(x) = x^n - A$ and using (5.1) the following particular iteration formula is obtained:

$$x^+ = x \cdot \frac{(n-1)(x)^n + (n+1)A}{(n+1)(x)^n + (n-1)A}. \tag{5.2}$$

In his paper, Carlson has shown that this formula holds for both even $n = 2m$ and odd $n = 2m+1$ roots. The method can be applied to approximation of fractional capacitors of the form $(1/s)^{1/n}$ in the following way:

$$\begin{aligned} G^+(s) &= G(s)\frac{(n-1)\,(G^n(s)) + (n+1)\,(H(s))}{(n+1)\,(G^n(s)) + (n-1)\,(H(s))}, \\ H(s) &= \frac{1}{s}, \quad G_0(s) = 1. \end{aligned} \tag{5.3}$$

This method was more recently considered in [100,132,137]. Since in this case the real variable A is replaced by the transfer function $H(s)$, convergence and rate of convergence cannot be evaluated in the same way as in the case of a real-valued function. Consider now an example.

Example 5.1 Using equation (5.3) we shall obtain an approximation of a fractional capacitor $\sqrt[5]{1/s}$. With two iterations the following transfer functions are obtained:

$$G_0(s) = 1, \tag{5.4}$$

$$G_1(s) = \frac{0.66667s(s+1.5)}{s(s+0.6667)}, \tag{5.5}$$

$$G_2(s) = \frac{G_{21}(s)}{G_{22}(s)}, \tag{5.6}$$

where

$$\begin{aligned} G_{21}(s) = {} & 0.4444s^7 + 9.062s^6 + 39.47s^5 + 77.81s^4 \\ & + 82.5s^3 + 47.75s^2 + 13.23s + 1, \end{aligned} \tag{5.7}$$

$$\begin{aligned} G_{22}(s) = {} & s^7 + 13.23s^6 + 47.75s^5 + 82.5s^4 \\ & + 77.81s^3 + 39.47s^2 + 9.063s + 0.4444. \end{aligned} \tag{5.8}$$

In Figure 5.1 the frequency response of the obtained approximation is shown. The response of the corresponding ideal fractional capacitor is also

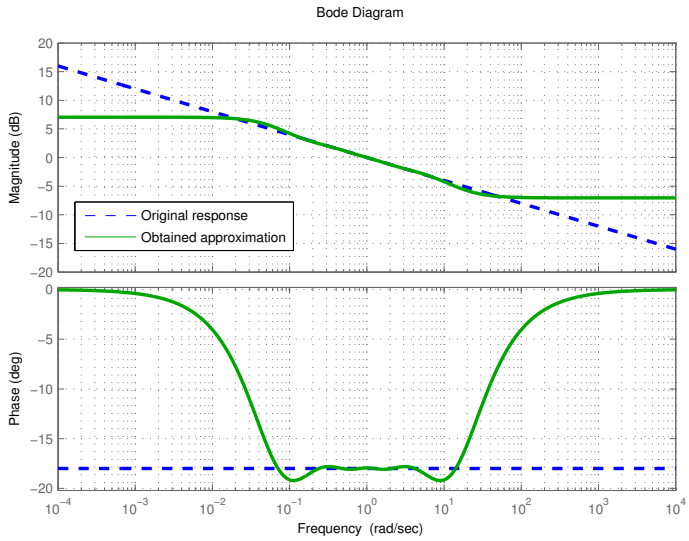


Figure 5.1: Frequency response of Carlson's approximation of the fractional capacitor $\sqrt[5]{1/s}$

given for comparison. It can be seen that the frequency range where the approximation is valid is quite narrow. It is possible to improve this result by increasing the number of formula iterations. However, in this case the order of the obtained rational transfer function may be very high.

This example illustrates that the method may not be very effective for approximating fractional differentiators and integrators on a wide frequency range. However, the method may be used for frequency-bounded implicit fractional transfer function approximation. In this work, we treat the case of a first-order fractional transfer function.

In general, a frequency-bounded non-integer differentiator/integrator may be represented by a first-order implicit fractional transfer function of the form

$$G(s) = \left(\frac{bs+1}{as+1}\right)^{\alpha}, \tag{5.9}$$

where $0 < \alpha < 1$. The frequency of the zero is in this case $\omega_z = 1/b$ and the frequency of the pole is $\omega_p = 1/a$, when $\alpha > 0$. Following the terminology in [21] and since in this case the transfer function has a single fractional power zero and a single fractional power pole, we also refer to this form as a Fractional Power Zero-Pole (FPZP) pair. Recall that this model corresponds to a FO lead-lag compensator in (2.30).

We now describe the algorithm, which can be used to obtain accurate approximations in form of zero-pole distributions for the fractional transfer function in (5.9).

Based on the discussion above, several problems of the original algorithm in [19] may be outlined:

- The initial estimate for approximation problem is not addressed;
- The method only allows to obtain approximations for transfer functions of order $1/n$;
- Resulting approximations can be of a very high order;
- The limited frequency range where the approximation is valid.

The specific application of Carlson's method could be different. In fact, when applied to the problem of approximating the transfer function in (5.9) for a limited frequency range, the algorithm provides very accurate results. Further, we describe the refined algorithm, which aims to solve the aforementioned problems.

First, we consider the initial estimate problem. Using the iteration formula (5.3) results in a recursive distribution of zeros and poles around a central frequency. In case of the fractional power zero-pole pair transfer function, this frequency is the geometric mean computed from the zero and pole frequencies such that

$$\omega_m = \sqrt{\omega_z \omega_p} = \frac{1}{\sqrt{ab}}. \tag{5.10}$$

It relates to the initial estimate choice through the magnitude of the fractional transfer function obtained at this frequency:

$$G_0(\omega) = |G(j\omega_m)| = \left| \frac{jb\omega_m + 1}{ja\omega_m + 1} \right|^{\alpha}. \tag{5.11}$$

When selecting the initial estimate according to (5.11) the resulting zero-pole distribution is then centered around ω_m ensuring that way the validity of the approximation around this frequency. When the ratio a/b is small, only two iterations are usually required to achieve a good result in the full frequency range.

The problem of approximating transfer functions of arbitrary real order using this method is much more difficult to solve. Here, we must choose a balance between accuracy and efficiency, since in case of order $1/n$ each iteration step involves computing the nth power of a transfer function obtained in the previous step. The order of the approximation grows rapidly. Thus, until a different, more efficient iteration formula is developed, we limit the resolution to 1/10. This allows to obtain approximations of orders accurate to at least one decimal place. However, there is no reason why a class of arbitrary orders could not be considered as well.

The problem of using the method to obtain an approximation for an arbitrary real α falls under the Egyptian fraction decomposition class of problems, i.e. an order α is decomposed into k simple fractions $1/m_k$:

$$\alpha = \frac{1}{m_1} + \frac{1}{m_2} + \cdots + \frac{1}{m_k}, \tag{5.12}$$

where $m_k \in \mathbb{N}$. The order decomposition algorithm is depicted in Figure 5.2 and is discussed below.

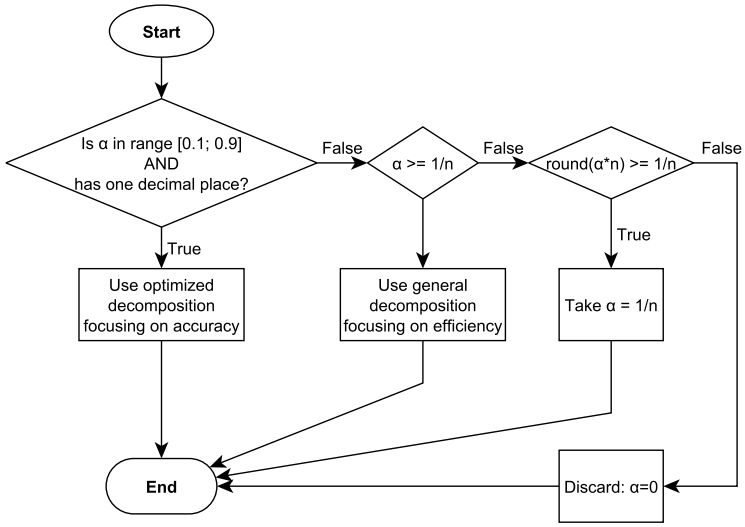


Figure 5.2: Order α decomposition algorithm

The optimized decomposition is conducted using fractions 1/2 (most efficient), 1/5 and 1/10 (accuracy consideration). The decimal fractions are then decomposed as follows:

$$\begin{aligned} 0.1 &= \tfrac{1}{10}, & 0.2 &= \tfrac{1}{5}, & 0.3 &= \tfrac{1}{5} + \tfrac{1}{10}, \\ 0.4 &= 2 \cdot \tfrac{1}{5}, & 0.5 &= \tfrac{1}{2}, & 0.6 &= 3 \cdot \tfrac{1}{5}, \\ 0.7 &= \tfrac{1}{2} + \tfrac{1}{5}, & 0.8 &= 4 \cdot \tfrac{1}{5}, & 0.9 &= 4 \cdot \tfrac{1}{5} + \tfrac{1}{10}. \end{aligned}$$

The fractional transfer function is then approximated as

$$G^{\alpha}(s) = \prod_{j=1}^{k} G_{base}^{\frac{1}{m_j}}(s), \tag{5.13}$$

where

$$G_{base}(s) = \frac{bs+1}{as+1}. \tag{5.14}$$

Note that the initial estimate is computed for every approximation of $G_{base}^{1/m_j}(s)$.

The general decomposition algorithm is given in Figure 5.3. In our case $M = 10$ and thus for $0 < \alpha < 1$ a decomposition will always be found, since

$$\sum_{k=2}^{10} \left(\frac{1}{k} \right) > 1. \tag{5.15}$$

For $\alpha > 1$ the general commutative property of a fractional operator is considered, so the approximation is found such that

$$G^{\alpha}(s) = G^{n}(s) \cdot G^{\gamma}(s), \tag{5.16}$$

where $n = \alpha - \gamma$ denotes the integer part of α and $G^{\gamma}(s)$ is obtained using (5.13). For the case when $\alpha < 0$, the approximation is

$$G^{-\alpha}(s) = \left(\frac{1}{G(s)} \right)^{\alpha}. \tag{5.17}$$

```
procedure ModelDecomposition(G, α, M)
    for P = 2 to M do
        if α ⩾ (1/P) then
            G ← G · G^{1/P}
            α ← α − (1/P)
        end if
    end for
end procedure
```

Figure 5.3: General decomposition algorithm

Finally, we address the problem of approximation order. We propose two possibilities for order reduction:

1. Reduction of matching zeros and poles;

2. Applying a balancing reduction technique, e.g., [134].

The first method may be invoked on each step of iteration when the order α is small to improve performance. The second method can be applied to the resulting approximation.

We conclude this section by noting the similarities in the approaches to realization of the fractional transfer function in (5.9) found in this work and in [21, 89]. Also, a similar implementation can be found in [132].

Therefore, it is possible to obtain the fractional differentiator/integrator approximations in the desired frequency range $\omega = [\omega_z; \omega_p]$ by selecting $b = \frac{1}{\omega_z}$, $a = \frac{1}{\omega_p}$ and using the following equation:

$$s^\alpha \approx a^\alpha G(s), \tag{5.18}$$

where $\alpha > 0$ corresponds to a fractional-order differentiator, $\alpha < 0$ corresponds to a fractional-order integrator and $G(s)$ is the approximation obtained using the above algorithm.

Consider now two examples illustrating the potential use of the described method.

Example 5.2 We shall obtain an approximation for the following implicit transfer function:

$$G_1(s) = \left(\frac{0.137s + 1}{15.294s + 1}\right)^{-1.115}. \tag{5.19}$$

After two iterations, the approximation is obtained. The minimal realization thereof is given by

$$\tilde{G}_1(s) = 187.95\frac{\tilde{G}_{1z}(s)}{\tilde{G}_{1p}(s)}, \tag{5.20}$$

where

$$\begin{aligned}\tilde{G}_{1z}(s) \;=\; & (s + 0.5746)(s + 0.1175)(s + 0.06539)(s + 3.504) \\ & (s^2 + 1.384s + 0.4795)(s^2 + 1.378s + 0.4802) \\ & (s^2 + 1.355s + 0.4829)(s^2 + 1.205s + 0.502),\end{aligned} \tag{5.21}$$

and

$$\begin{aligned}\tilde{G}_{1p}(s) \;=\; & (s + 7.299)(s + 4.062)(s + 0.8307)(s + 0.1362) \\ & (s^2 + 1.378s + 0.475)(s^2 + 1.369s + 0.4743) \\ & (s^2 + 1.339s + 0.4717)(s^2 + 1.146s + 0.4538).\end{aligned} \tag{5.22}$$

The order error is $\epsilon = 0.0039$. The comparison of the ideal response and the response of the obtained approximation is given in Figure 5.4.

Example 5.3 In this example we shall implement a fractional lead compensator, discussed in [72]. Consider a transfer function that describes a position servo:

$$G_2(s) = \frac{1.4}{s(0.7s + 1)}\mathrm{e}^{-0.05s}. \tag{5.23}$$

Based on some performance specifications (phase margin $\varphi_m = 80°$ and gain crossover frequency $\omega_c = 2.2$ rad/s), the controller was proposed such that

$$C_2(s) = \left(\frac{2.0161s + 1}{0.0015s + 1}\right)^{0.702}. \tag{5.24}$$

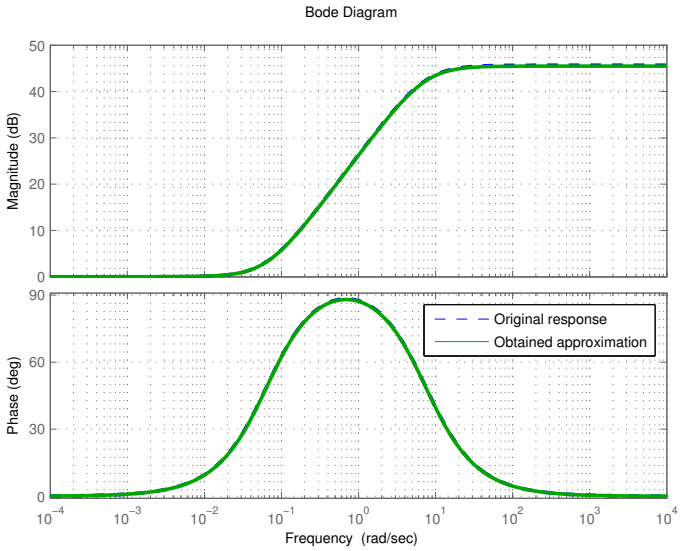


Figure 5.4: $\tilde{G}_1(s)$ approximation frequency response vs. $G_1(s)$ ideal frequency response

The order of the model approximated in 3 iterations is 328. Using the balanced realization technique with the target order model 5 results in a suitable approximation of the fractional transfer function given by

$$\tilde{C}_2 = 154.65\frac{\tilde{C}_{2z}}{\tilde{C}_{2p}}, \tag{5.25}$$

where

$$\tilde{C}_{2z} = (s+292.4)(s+76.33)(s+17.01)(s+3.141)(s+0.6064), \tag{5.26}$$

and

$$\tilde{C}_{2p} = (s+590.9)(s+199.6)(s+49.45)(s+10.48)(s+1.829). \tag{5.27}$$

The resulting control system open-loop frequency response $\tilde{C}_2(j\omega)G_2(j\omega)$ is shown in Figure 5.5. It can be seen that the design specifications are correctly fulfilled.

5.2 Efficient Analog Implementation of Fractional-order Models and Controllers

In this section we provide an overview of fractance circuit approximation methods. Then, the unified approach to generation and application of such circuits is described.

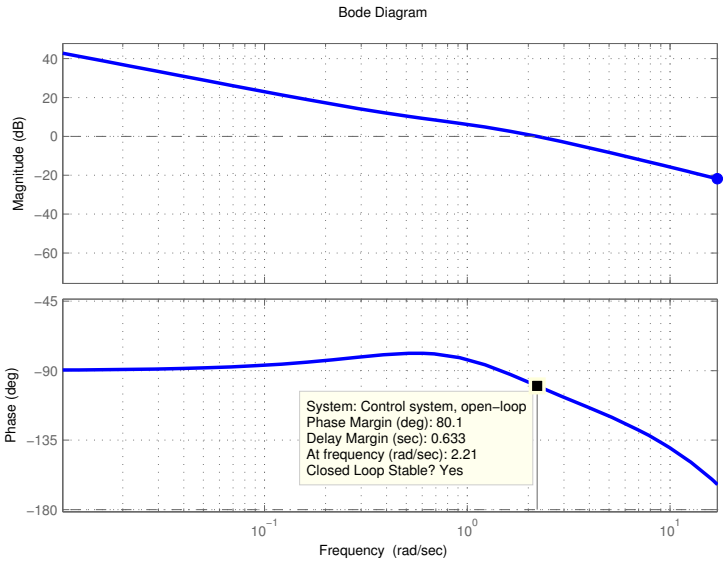


Figure 5.5: Control system open-loop frequency response

5.2.1 Approximation Methods

In the following, we provide a summary of network structures and corresponding synthesis methods used in this work with relevant comments.

Over the years, several methods involving use of Cauer and Foster canonical network forms were proposed [100]. Two of these structures will be used in this work. The Cauer I form RC circuit, presented in Figure 5.6, has the following impedance, obtained by applying continued fraction expansion [22]:

$$Z_{RC}(s) = R_1 + \cfrac{1}{C_2 s + \cfrac{1}{R_3 + \cfrac{1}{C_4 s + \cdots}}}. \tag{5.28}$$

The Foster II form RC circuit is given in Figure 5.7. The corresponding admittance can be expressed by means of partial fraction expansion in the following way:

$$Y_{RC}(s) = \frac{1}{Z_{RC}(s)} = \frac{1}{R_p} + C_p s + \sum_{i=1}^{n} \frac{K_i s}{s + \sigma_i}, \tag{5.29}$$

where $K_i = 1/R_i$ and $\sigma_i = K_i/C_i$.

In general, in order to obtain the fractance network component values there are several choices:

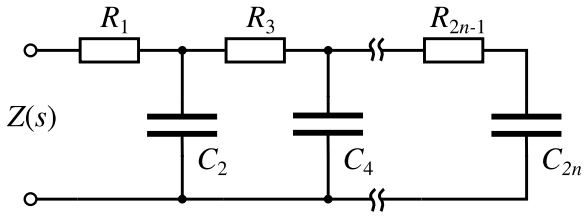


Figure 5.6: Cauer I form canonical RC network

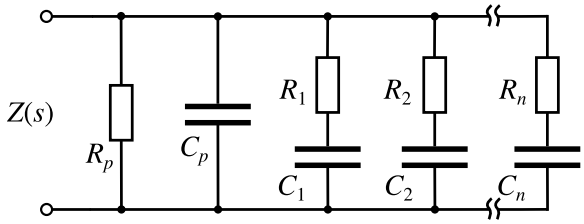


Figure 5.7: Foster II form canonical RC network

- Use a suitable approximation technique to obtain the impedance in form $Z(s)$ and develop it into a suitable expansion, thus obtaining the R, C, or L component values. The values of the components are not well-scaled and negative values may be obtained, in which case one would need to use negative impedance converters [100].
- Use constrained optimization to identify the network.
- Apply a method to derive the required component values directly in a much more controlled manner.

In [75], a RC driving-point immitance approach, applied to a Foster II canonical form RC network, was proposed to achieve a constant phase angle φ over a designated frequency range. Useful relations were highlighted to determine the values of the R and C subsequent components. A similar approach was used in [60] and later in [27,28,133]. In what follows, we briefly describe this method and provide its application to fractional-order system and controller implementation.

Given generation parameters α (fractional operator order), $\Delta\varphi$ (phase ripple), R_1 and C_1 (base resistor and capacitor values, which are chosen according to the frequency range of interest). According to these values, the following parameters are calculated:

$$\eta \approx \frac{0.24}{1+\Delta\varphi}, \quad a = 10^{\alpha \log(\eta)}, \quad b = \frac{\eta}{a}, \tag{5.30}$$

where $0 < a < 1$ and $0 < b < 1$. The values of m resistors and m capacitors, comprising the network, are then obtained by using the following set of

synthesis formulae:

$$R_k = R_1 a^{k-1}, \quad C_k = C_1 b^{k-1}, \quad k = 1, 2, \ldots, m. \tag{5.31}$$

The values of R_p and C_p are obtained using

$$R_p = R_1 \frac{1-a}{a}, \quad C_p = C_1 \frac{b^m}{1-b}. \tag{5.32}$$

Finally, we need to take into account the correction gain K. If the approximated operator of order α is given by an integer-order transfer function $G(\alpha, s)$ and the approximation yields an admittance $Y(j\omega)$ in (5.29), then at a frequency

$$\omega_m = \frac{\sqrt{a}}{R_1 C_1 \eta^{\lfloor m/2 \rfloor - 1}}, \tag{5.33}$$

where $\lfloor \cdot \rfloor$ denote the floor function, the gain is computed as

$$K = \frac{|G(\alpha, j\omega_m)|}{|Z(j\omega_m)|}, \tag{5.34}$$

where $Z(j\omega_m)$ is the impedance of the obtained approximation.

This method can be effectively applied to implement a FOPID controller. We may use the synthesis procedure in (5.30)–(5.34) separately for the fractional integrator and fractional differentiator to arrive at two fractance networks, which will form the controller in a general active filter configuration as illustrated in Figure 5.8. $Z_1(s)$ and $Z_2(s)$ should be chosen accordingly and reduced to a trivial resistance if need be.

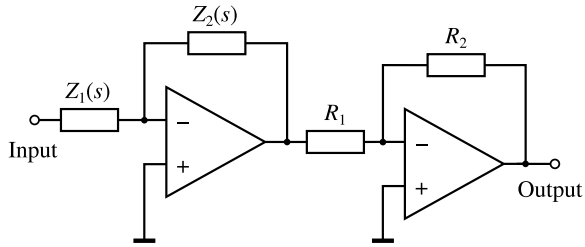


Figure 5.8: Analog approximation of a fractional-order operator

We conclude this section by summarizing the steps required to implement a fractional lead-lag compensator. The same synthesis formulae can be used, however:

- The frequency range of approximation, determined by $\tau = R_1 C_1$, must be correctly chosen;

- The gain of the differential or integral component must be corrected by a factor $1/\omega_z^\alpha$ such that

$$G_c(\alpha, s) = \omega_z^{-\alpha} G(\alpha, s), \tag{5.35}$$

where ω_z is the zero frequency in (2.30), $G_c(\alpha, s)$ and $G(\alpha, s)$ are integer-order transfer functions, approximating the fractional lead-lag compensator and the corresponding fractional-order operator, respectively.

5.2.2 Unified Approach to Fractance Network Generation

It is our goal to obtain a general enough approach to systematization of existing network topologies and their generation. The general framework for fractance network synthesis may be implemented programmatically using a personal computer. The network generation and analysis may be handled by means of a central component—an object in an object-oriented programming language, containing complete circuit information. The object contains references to

- Particular network structures, however complex, which return computed network transfer functions (impedance values);
- Corresponding implementations.

The idea is that a single structure can have several different implementations, including, e.g., optimization based ones. The relations are illustrated in Figure 5.9, so that the best one may be chosen for a particular application.

This way one can implement, e.g., a FOPID controller, and store it in a single object. For generality, we also consider the possibility of using inductive components in network structures.

Fractance is implemented by means of a class `frac_rcl()`, which is shown in an UML diagram in Figure 5.10 and has following properties:

- *model* — a model of the fractional-order system (`fotf` object);
- *structure* — network structure (Cauer, Foster, etc.);
- *implementation* — function that carries out the actual computation of the network component values;
- ω — frequency points used for model validation;
- *params* — parameters used for implementation and/or in the structure;
- K — network gain compensation factor(s);

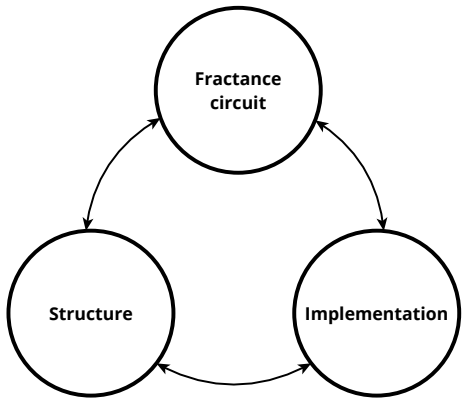


Figure 5.9: Fractance, network structure, and implementation relations

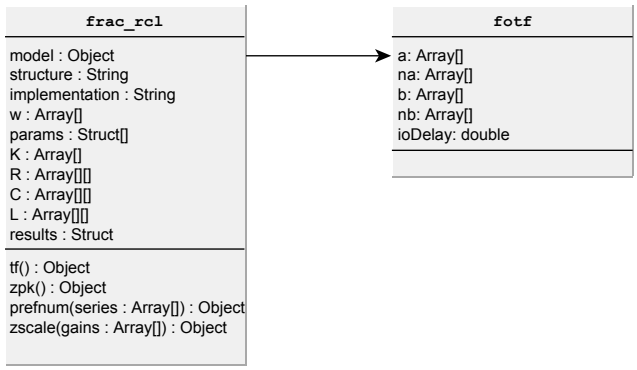


Figure 5.10: UML diagram of the *frac_ rcl* class in relation to the *fotf* class

- R, C, L — cell array with component value vectors, the size of the array is determined by the number of substructures within the main structure;
- *results* — implementation/validation results.

The first five parameters are used to create the `frac_rcl()` object. The following particular methods may be implemented:

- `tf()`, `zpk()` — return the impedance $Z(s)$ in transfer function or zero-pole-gain format, corresponding to the fractance circuit, for network analysis. This can be used to automate, e.g. frequency response analysis.
- `prefnum()` — locates closest component values according to the preferred series and replaces network components accordingly. It is also

possible to provide custom values of components so that the algorithm will choose the closest matches among a custom set of values. This can be necessary to analyze the changes of the system frequency characteristics due to variation of component values.

- `zscale()` — implementation of impedance scaling used to shift the values of discrete electronic components into the feasible domain.

Abstraction of fractance into a class has obvious benefits. The user has an option of writing implementations for existing structures, or using the provided ones, while still using a single universal object to encompass the network. Some specific function naming schemes and coding conventions need to be followed by the programmer. Also, the programmer is responsible for documenting new, custom structures.

Relevant examples are provided in Chapter 6, where a MATLAB environment implementation of the proposed framework is available as part of the FOMCON toolbox.

5.3 Digital Implementation of Fractional-order Controllers

In the following, the description of the method used for digital implementation of the FO controllers is provided. Realizations of FOPID controller and FO lead-lag compensator are discussed.

5.3.1 Discrete-time Oustaloup Filter Approximation for Embedded Applications

Recall the Oustaloup approximation method from Section 2.3. Given the approximation frequency range $[\omega_b, \omega_h]$ rad/s, order of approximation $\nu \in \mathbb{Z}^+$ and fractional power $\alpha \in [-1, 1] \subset \mathbb{R}$, we proceed to compute $(2\nu + 1)$ zeros and $(2\nu + 1)$ poles of the filter as

$$\omega'_k = \omega_b \theta^{\frac{(k+\nu+0.5-0.5\alpha)}{2\nu+1}}, \ \omega_k = \omega_b \theta^{\frac{(k+\nu+0.5+0.5\alpha)}{2\nu+1}}, \tag{5.36}$$

where $k = \{-\nu, -\nu+1, \ldots, 0, \ldots, \nu-1, \nu\}$ and $\theta = \omega_h/\omega_b$. Thus the continuous recursive Oustaloup filter transfer function is obtained in the form

$$\hat{G}(s) = \omega_h^{\alpha} \frac{(s-\omega'_{-\nu})(s-\omega'_{-\nu+1}) \cdots (s-\omega'_{\nu})}{(s-\omega_{-\nu})(s-\omega_{-\nu+1}) \cdots (s-\omega_{\nu})}. \tag{5.37}$$

The filter approximates a fractional-order operator

$$s^{\alpha} \approx \hat{G}(s) \tag{5.38}$$

in the chosen frequency range. The amount of ripple in the phase response of this filter can also be determined [89]. In the following, we describe the discretization method which, when employed, can serve as a basis for discrete-time Oustaloup filter generation and may be implemented on an embedded device.

Suppose that we are given a sampling interval $T_s \in \mathbb{R}^+$. Then we may set the higher frequency bound of approximation in (5.36) to $\omega_h = 2/T_s$. Next, consider the zero-pole matching equivalents method for obtaining a discrete-time equivalent of a continuous time transfer function [35]. The following mapping is used for both zeros and poles:

$$z = \mathrm{e}^{sT_s}, \tag{5.39}$$

where s denotes a particular zero or pole. Therefore, for each k in (5.36) we take

$$\sigma'_k = \mathrm{e}^{-T_s\omega'_k}, \quad \sigma_k = \mathrm{e}^{-T_s\omega_k} \tag{5.40}$$

thus mapping continuous zeros and poles to their discrete-time equivalents directly. We notice that once the mapping is done, we need to compute the gain of the resulting discrete-time system at the central frequency $\omega_u = \sqrt{\omega_b\omega_h}$. This can be done by first finding the gain of the resulting discrete-time system by taking

$$K_u = \left|H(\mathrm{e}^{j\omega_u T_s})\right|. \tag{5.41}$$

We also know the correct gain at this frequency

$$K_s = \omega_u^\alpha. \tag{5.42}$$

So, finally we obtain the gain of the system as

$$K_c = K_s/K_u. \tag{5.43}$$

The discrete-time system is thus described by a transfer function of the form

$$H(z) = K_c \frac{(z-\sigma'_{-\nu})(z-\sigma'_{-\nu+1})\cdots(z-\sigma'_{\nu})}{(z-\sigma_{-\nu})(z-\sigma_{-\nu+1})\cdots(z-\sigma_{\nu})}. \tag{5.44}$$

Due to the fact that the order of the approximated operator is $\alpha \in \mathbb{R}^+$ the transfer function in (5.37) is stable [89] and the corresponding discrete-time equivalent (5.44) is also stable.

Next we address the problems associated with implementing the generation scheme described above on an embedded device, such as a microcontroller. We have to take the following into consideration:

- Performance limitations;
- Limited computational abilities;

- Potential memory size limitations.

The first item completely depends on the type of microprocessor (and potentially additional computational hardware units) used in the implementation.

We notice that (5.41) involves computations with complex numbers. However, we can compute a particular factor $(z-\sigma)$ in (5.44) at the frequency ω_u as follows

$$\left|\mathrm{e}^{j\omega_u T_s} - \sigma\right| = \left|\cos(\omega_u T_s) + j\sin(\omega_u T_s) - \sigma\right| = \\ = \sqrt{1 - 2\sigma\cos(\omega_u T_s) + \sigma^2} \quad (5.45)$$

due to Euler's formula. Therefore, the gain of the system specified by discrete-time zeros and poles in (5.40) may be computed as

$$K_u = \frac{\prod_{k=-\nu}^{\nu} \left(1 - \sigma'_k\theta + (\sigma'_k)^2\right)^{0.5}}{\prod_{k=-\nu}^{\nu} \left(1 - \sigma_k\theta + \sigma_k^2\right)^{0.5}}, \quad (5.46)$$

where $\theta = 2 \cdot \cos(\omega_u T_s)$ is constant at the given frequency ω_u and sampling interval T_s which needs to be computed only once. After computing this gain one arrives at the final gain K_c of the discrete-time approximation by using equation (5.43).

This system can be implemented as an IIR filter. The next step is to transform this representation into second-order section form to improve computational stability. Consider the set of discrete-time zeros (poles) that we have obtained earlier

$$z = \left\{\sigma_{-\nu}, \sigma_{-\nu+1}, \ldots, \sigma_0, \ldots \sigma_\nu, \sigma_{\nu-1}, \sigma_\nu\right\}. \quad (5.47)$$

Due to the generation method (5.36) the set in (5.47) is an ordered set. In order to arrive at the second-order section form for the zero (pole) polynomial we proceed as follows. We have $2\nu+1$ zeros (poles), so there are $\nu+1$ second-order sections (including a single first-order section). Therefore, we have the polynomial

$$h(z) \quad = \quad (1 - \sigma_\nu z^{-1}) \cdot \prod_{k=0}^{\nu-1} \zeta(z) \quad (5.48)$$

in the variable z, where $\zeta(z) = 1 + (c_k + d_k)\, z^{-1} + (c_k \cdot d_k)\, z^{-2}$, $c_k = -\sigma_{-\nu+2k}$ and $d_k = -\sigma_{-\nu+2k+1}$. So finally we arrive at the form

$$H(z) = K_c \prod_{k=1}^{\nu} \frac{1 + b_{0k} z^{-1} + b_{1k} z^{-2}}{1 + a_{0k} z^{-1} + a_{1k} z^{-2}}, \quad (5.49)$$

which can be effectively used as an IIR filter in control applications.

We now turn to the issue of storing the aforementioned coefficients on an embedded system with dynamic memory. Additionally we consider the necessary memory size for digital signal processing related computations.

While it is usually possible to use dynamic memory allocation for both zero/pole generation and SOS coefficient arrays, on embedded systems with limited memory size it is safer to use a static memory allocation architecture to circumvent potential run-time problems arising from, e.g., memory fragmentation. Therefore, care must be taken to choose a sufficient maximal approximation order ν_{max} and preallocate the necessary array memory space beforehand. Suppose that a floating-point data type with a size of ψ bytes is available in a particular implementation. In the following, we provide some computations related to minimal memory requirements. First, to store arrays of values for discrete zero/pole calculation:

$$\text{Memory for zero/pole arrays} = 2\psi(2\nu_{max} + 1)\,\text{bytes}. \tag{5.50}$$

Now we provide the memory requirements for second-order section coefficient storage. Note that in (5.49) we only need to store coefficients b_1, b_2, a_1 and a_2. Then for both arrays we have

$$\text{Memory for SOS arrays} = 4\psi\chi(\nu_{max} + 1)\,\text{bytes}, \tag{5.51}$$

where χ is the number of approximated operators. Finally, we will need memory elements for the digital signal processing application:

$$\text{Memory for DSP} = 2\psi\chi(\nu_{\max} + 1)\,\text{bytes}. \tag{5.52}$$

The total amount of memory required for the arrays is thus

$$\text{Total memory} = 2\psi\left((3\chi + 2)\nu_{max} + 3\chi + 1\right)\text{ bytes}. \tag{5.53}$$

Example 5.4 Consider the Atmel AVR ATmega8 microcontroller, which we use as the basis for the implementation of a digital filter approximating a fractional-order operator with an order α such that $0 < \alpha < 1$. Suppose that a single precision floating-point data type is available. Then $\psi = 4$ and $\chi = 1$ and for $\nu_{max} = 10$ we need to preallocate

$$2 \cdot 4\left((3 \cdot 1 + 2) \cdot 10 + 3 \cdot 1 + 1\right) = 432\,\text{bytes},$$

which takes up 42.18% out of 1024 bytes of SRAM memory of this particular microcontroller.

We remark that it is possible to reuse some static memory blocks during the generation of the coefficients thus reducing the necessary memory size requirements.

5.3.2 FOPID Controller Implementation

Digital implementation of the fractional-order PID controller may be obtained as

$$H_{PI^{\lambda}D^{\mu}}(z) = K_p + K_i H_I^{-\lambda}(z) + K_d H_D^{\mu}(z^{-1}), \tag{5.54}$$

where K_p, K_i, and K_d are gains of the parallel form of the controller as in (2.28), $H_I^{\lambda}(z)$ corresponds to a discrete-time approximation of a fractional-order integrator of order λ and $H_D^{\mu}(z)$ corresponds to a discrete-time approximation of a fractional-order differentiator of order μ, such that $\lambda, \mu \in [0, 1]$.

Next, we address the issue of implementing the fractional-order integrator. Recall the discussion in Section 2.4. Due to (2.29) we should implement the integrator as

$$H_I(z) = H^{1-\lambda}(z) \cdot H_I(z), \tag{5.55}$$

where $H^{1-\lambda}(z)$ is computed using the method presented above, and

$$H_I(z) = \frac{T_s}{(1 - z^{-1})} \tag{5.56}$$

is the discrete-time integrator.

5.3.3 FO Lead-Lag Compensator Implementation

Recall the fractional-order lead-lag compensator from (2.30) rewritten as

$$C_L(s) = K_L \left(\frac{b_L s + 1}{a_L s + 1} \right)^{\alpha_L}. \tag{5.57}$$

To implement this controller one must choose the appropriate approximation frequency bounds ω_b and ω_h in (5.36) such that

$$\omega_b = 1/b_L, \quad \omega_h = 1/a_L. \tag{5.58}$$

In addition, a correction gain $K_c = b_L^{\alpha}$ must be applied to the Oustaloup filter approximation. The approximation is then given by

$$\hat{C}_L(s) = K_L K_c \hat{G}(s), \tag{5.59}$$

where $\hat{G}(s)$ is computed in (5.37). It can be easily deduced that a fractional lag compensator corresponds to a I^{λ} controller with

$$K_i = K_L K_c, \quad \lambda = \alpha_L \tag{5.60}$$

and a fractional lead compensator corresponds to a D^{μ} controller with

$$K_d = K_L K_c, \quad \mu = \alpha_L \tag{5.61}$$

with the parameters ω_b and ω_h chosen according to the parameters b_L and a_L. It should be noted that this method, as well as the choice of appropriate frequency bounds in (5.58), works only in case of the original Oustaloup filter in (5.36), not in case of the modified filter discussed in [72, 146].

5.3.4 Controller Reset Logic

It is also necessary to address the state reset logic for the IIR filters that are used to implement the fractional-order controllers. Denote by $e(k)$ the kth sample of the error signal $e(\cdot)$. We propose the following basic filter memory reset logic based on the notion of a maximal error change rate margin ρ. The reset condition is expressed as follows

$$|e(k) - e(k-1)| > \rho. \tag{5.62}$$

Thus if the controller detects a sudden change in the error signal, IIR filter and integer-order integrator memory will be cleared, yielding zero initial conditions for the whole fractional-order PID controller or lead-lag compensator. It is important to select the value of the change rate margin ρ well above measurement noise or potential disturbance level.

Particular examples of digital controller implementation are provided in the next section.

5.4 Experimental Platform for Real-Time Closed-Loop Simulations of Control Systems

In the following, a platform for real-time hardware-in-the-loop control experiments is described.

In terms of hardware, we use two different data acquisition devices:

- ATDaqV2 data acquisition device developed in Alpha Control Laboratory [119]. It is connected to a personal computer through either RS232 or USB interfaces and is based on a serial protocol. It offers the following features [124]:
  - Analog channels: single-ended, 2 inputs and 2 outputs;
  - Sample resolution on all channels: 12 bit;
  - Sample rate: up to 10kSPS in single channel, half-duplex mode (input or output); up to 2.5kSPS in dual channel, full-duplex mode (suitable for closed-loop simulations) with single-sample delays;
  - Input/output type and range: voltage, 0...5V;
  - Reference voltage: on-board.
- RT-DAC/USB2 data acquisition device offered by INTECO—a multifunction analog and digital I/O board dedicated to real-time data acquisition and control in the Windows operating system environment and offering the following features in the analog signal section [47]:

– Analog channels: single-ended, 16 inputs, and 4 outputs;
– Sample resolution: 12 bit for input channels (with programmable gain); 12 or 14 bit on output channels;
– Conversion time for analog channels: 5.4μs;
– Settling time for output channels: 10μs;
– Input/output type and range: voltage, $\pm$10V on input and output channels, or $\pm$5V on output channels with enhanced resolution;
– Reference voltage: on-board.

The use of the ATDaqV2 device is preferred with the basic version of the controller prototype, while the more advanced RT-DAC is used in experiments with a more sophisticated controller prototype, both of which are described in Section 5.5.

The general schematic depicting the hardware connection between the devices considered in the experiments is given in Figure 5.11. The real-life controller prototype is interfaced with the personal computer via a data acquisition (DAQ) board, which can be one of the above.

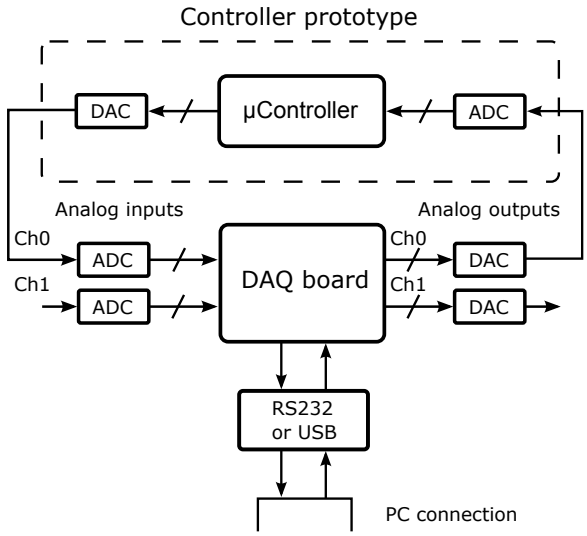


Figure 5.11: Real-time hardware-in-the-loop platform for control experiments

In terms of software, we use the MATLAB/Simulink environment with Real-Time Windows Target toolbox [65]. In this work we consider experiments, where the controller prototype is externally connected to a MATLAB/Simulink based simulation running in real time on a personal computer through an analog interface. This allows to verify and evaluate the performance of the prototype in case of arbitrary complex models of control objects.

In addition, we consider experiments, where the MATLAB/Simulink environment serves only as an additional interface between the controller and the controlled object.

5.5 Development of a Hardware FOPID Controller Prototype

For the purposes of verification of the developed FOPID control and design algorithms, a hardware implementation of FOPID controller is necessary. In this work, microcontroller based realizations are considered. The firmware is written in C language which makes the resulting code easily portable across different microcontroller families.

Hereinafter, we consider 8-bit Atmel AVR [8] and 32-bit STMicroelectronics STM32F407 [109] microcontroller families for the digital implementation of the FOPID controller prototype.

5.5.1 Atmel AVR Microcontroller Family based Implementation

The choice of using Atmel AVR 8-bit microcontroller family for the implementation of the FOPID controller is due to the following highlights of the Atmel AVR line [11].

- Outstanding flash memory technology;
- Single-cycle instruction execution;
- Wide variety of operating voltages;
- Architecture designed for the C language;
- One set of development tools for the entire AVR line;
- In-system programming, debugging, and verification capability;
- World-wide interest in the AVR microcontroller line.

The last item is seen to be of fairly high importance, since the availability of FOPID control code for AVR microcontrollers may potentially influence the growing spread of FO technology in both academia and industry.

The developed prototype has the structure depicted in Figure 5.12. The prototype consists of a controller board, an input-output board, and a power supply. The I/O board is plugged onto the controller board. Internally the ADC and DAC devices communicate with the microcontroller using the inter-integrated circuit (I^2C) interface. The particular tested microcontrollers include ATmega8A and ATmega324, whereby the choice of controller depends

on the application—for basic FOPID control the former is sufficient, for more advanced tasks, such as controller tuning, the latter must be used to meet the flash and dynamic memory requirements. The I/O board has the following features:

- Analog channel: single-ended, 1 input and 1 output;
- Sample resolution: 12 bit;
- Sample rate: up to 10kSPS;
- Input/output type and range: voltage, 0...5V;
- Reference voltage: on-board.

The described prototype may be used in particular embedded applications, e.g., temperature control. However, due to performance limitations in terms of floating-point computations only a single-precision data type is available. For practical FOPID control this will suffice, following the discussion in Section 5.3.

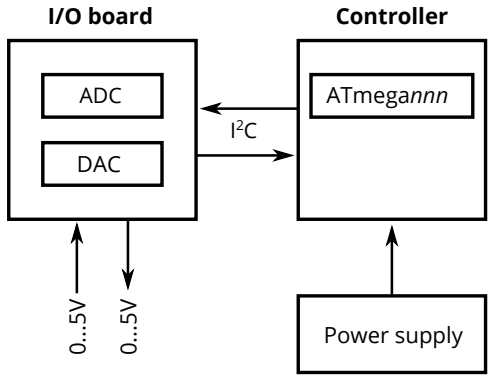


Figure 5.12: Structure schematic of the developed FOPID controller prototype based on Atmel AVR ATmega*nnn* microcontroller

In what follows, several examples related to FOPID controller approximation and tuning are provided.

Example 5.5 Consider the problem of obtaining a digital approximation of a FOPID controller with the parameters

$$K_p = k_1, \quad K_i = k_2, \quad K_d = k_3, \quad \lambda = q_1, \quad \mu = q_2. \tag{5.63}$$

The microcontroller board in Figure 5.12 is based on ATmega8A microcontroller clocked at 16MHz. Suppose that the suitable frequency range for an Oustaloup filter of order $\nu = 5$ is $\omega = [0.0001, 10]$ rad/s with $\nu_{max} = 10$. The sampling interval is $T_s = 0.2$s. Denote by τ_g and τ_s the time interval

that is required for controller generation and sample computation, respectively, under the conditions above. We have the following per the report of AVR Simulator: $\tau_g = 27.6224\,\text{ms}, \tau_s = 1.8904\,\text{ms}$. Obviously, the controller must compute the next output sample faster than the specified sampling rate. Thus, sampling rates up to $f_s \approx 500$ Hz are possible in this case. Note, however, that it takes much longer to compute the coefficients of the controller. This should be considered when the controller is running in a closed loop and, consequently, in autotuning applications. In Table 5.1 a summary of time requirements for controller generation and sample computation for $\nu = 6, 7, \ldots, 10$ is given.

Table 5.1: Time requirements for controller generation and sample computation for different Oustaloup filter orders

ν	τ_g [ms]	τ_s [ms]	Max.applicable f_s [Hz]
6	32.2914	1.9868	480
7	37.1326	2.0832	450
8	42.2011	2.1796	425
9	46.8712	2.2759	400
10	51.5617	2.3723	400

Example 5.6 In this example we illustrate the use of the fractional order sweep method from Section 4.5 on embedded hardware. For the purposes of implementing the FOPI controller, the Atmel AVR ATmega8A microcontroller is used. The equations and the power sweep method from Section 4.5 are implemented in C language and compiled using AVR-GCC. The following is a continuation of Example 4.3. The λ parameter sweep was done for the AMIGO PI controller. In terms of performance, our current software implementation requires approximately 34.6 million clock cycles to arrive at the solution $\lambda^\star = 0.8$, that is, around 4.3s with the microcontroller clocked at 8MHz. This does not include the time necessary to compute the gain margin, as it is not used in the cost function. Once λ is computed, the controller is generated. It is then verified using a real-time prototyping platform from Section 5.4.

The process model (4.109) is running in Simulink, while the controller prototype is connected to it externally via the DAQ board. The comparison of step responses of closed-loop control systems with the AMIGO PI and the achieved FOPI is given in Figure 5.13. A noticeable improvement in control quality can be observed.

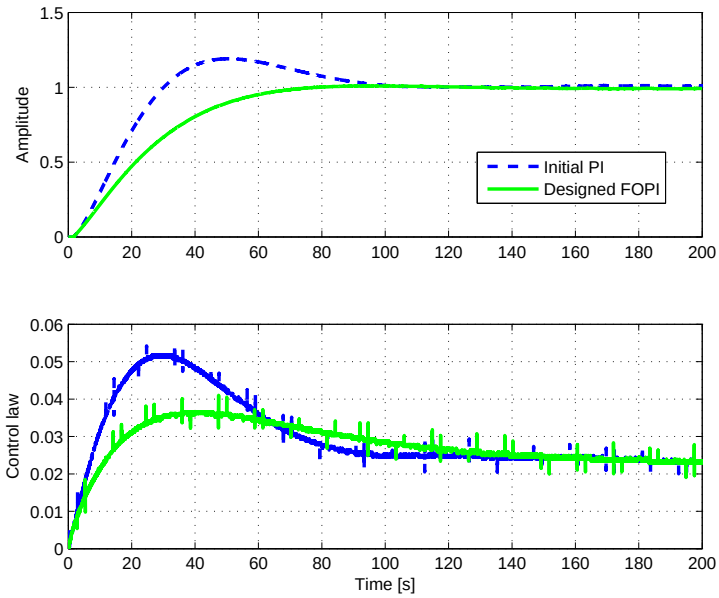


Figure 5.13: AMIGO PI vs. designed FOPI

Example 5.7 For this example, the algorithm in Example 4.4 is verified on the hardware prototype. This time the Atmel ATmega324P microcontroller is considered for the implementation of the FOPID controller. In the following, some initial benchmarking results are provided. The C code is compiled using AVR-GCC with optimization option "-O1". The same gains as in Example 4.4 were obtained in 4 algorithm iterations in approximately 1.66M clock cycles, or 83.37ms with the microcontroller clocked at 20MHz.

The resulting controller is also verified by means of pure software and hardware-in-the-loop (HIL) real-time simulations. Pure software simulations are done in MATLAB/Simulink environment. For the HIL part, the prototyping platform from Section 5.4 is used, where the process model (4.109) is running in Simulink, while the controller prototype is connected to it externally via the DAQ board.

The results of real-time simulations are shown in Figures 5.14 and 5.15. The gain of the plant is varying in the range $\pm 25\%$. The following observations can be made. First, as expected, the overshoot value does not change significantly between experiments, where the plant gain is different. This shows that the iso-damping property in (4.76) is indeed satisfied. Second, the corresponding results of pure software and HIL simulations are very close, which points to the adequate implementation of the controller prototype.

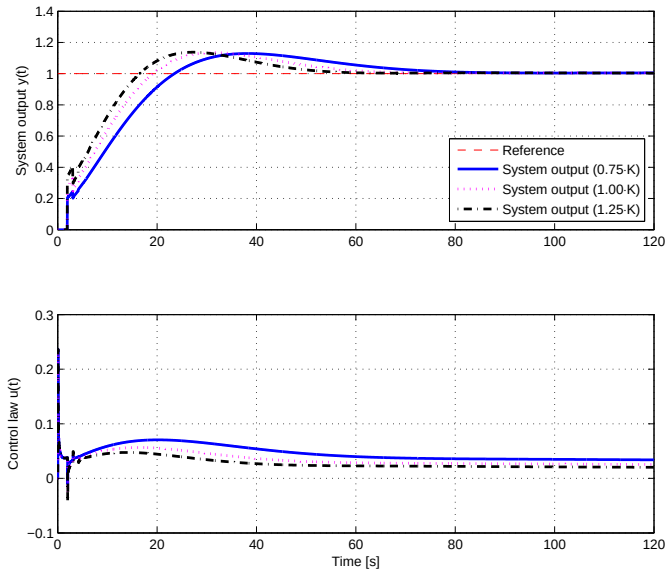


Figure 5.14: Pure software simulations of the FOPID control system with varying gain ($\tilde{K} = \{0.75K, 1.00K, 1.25K\}$)

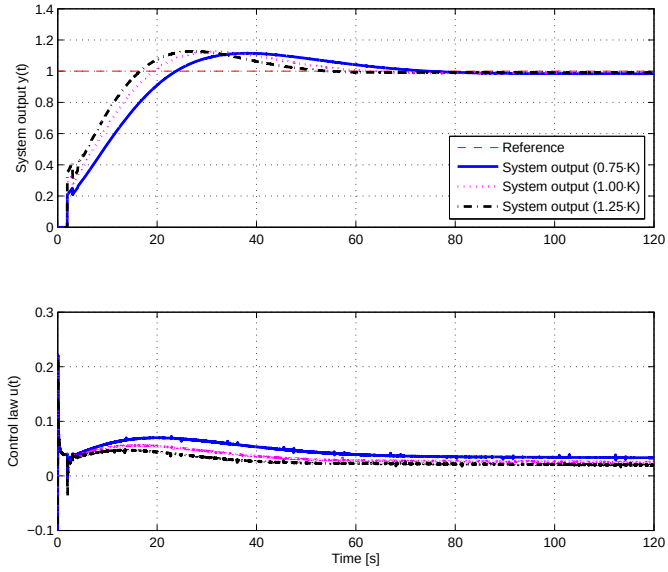


Figure 5.15: Hardware-in-the-loop simulations of the FOPID control system with varying gain ($\tilde{K} = \{0.75K, 1.00K, 1.25K\}$)

5.5.2 STMicroelectronics STM32F407 Microcontroller Family based Implementation

The STM32F407*nn* family of microcontrollers are based on the high performance ARM Cortex-M4 32-bit RISC core operating at a frequency up to 168MHz. The core has a dedicated floating-point unit (FPU) and implements a full set of DSP instructions. Therefore, these microcontrollers are natural candidates for complex digital signal processing applications.

In this work, the STM32F407VG microcontroller is used for the digital implementation of a FOPID controller. The developed hardware prototype has a modular structure depicted in Figure 5.16.

The controller module features the STM32F407VG microcontroller, which takes care of all necessary computations related to digital control, as well as drives the graphic display (resolution: 84x48 pixels) and the the input/output board through the serial peripheral interface (SPI). Manual controls allow to configure the controller prototype as needed.

The I/O board comprises analog-to-digital and digital-to-analog converters, as well as the Maxim Integrated MAX31855 chip providing a K-type thermocouple input [68] and the necessary signal conditioning circuits. The board has the following features:

- Analog channel: 2 inputs and 2 outputs:
  - Thermocouple input, supporting type K thermocouples;
  - Voltage input, range: ± 10 V;
  - Current loop output: $0(4) \ldots 20$ mA;
  - Voltage output: range ± 10 V;
- Sample resolution: 12 bit on alanog channels; 14 bit resolution of the thermocouple converter;
- Sample rate: up to 10kSPS;
- Reference voltage: precision; on-board.

The fully assembled prototype is shown in Figure 5.17. Both the controller and I/O board are mounted on a DIN rail. The prototype is designed mainly for laboratory experiments. For industrial use, a suitable form-factor and a suitable choice of components (e.g., manual controls and display) is necessary to meet industrial standards.

The examples from Section 5.5.1 are also applicable in case of this prototype, since the code is ported directly. However, the performance figures are obviously different. An example illustrating the use of this prototype is provided in Section 7.3.

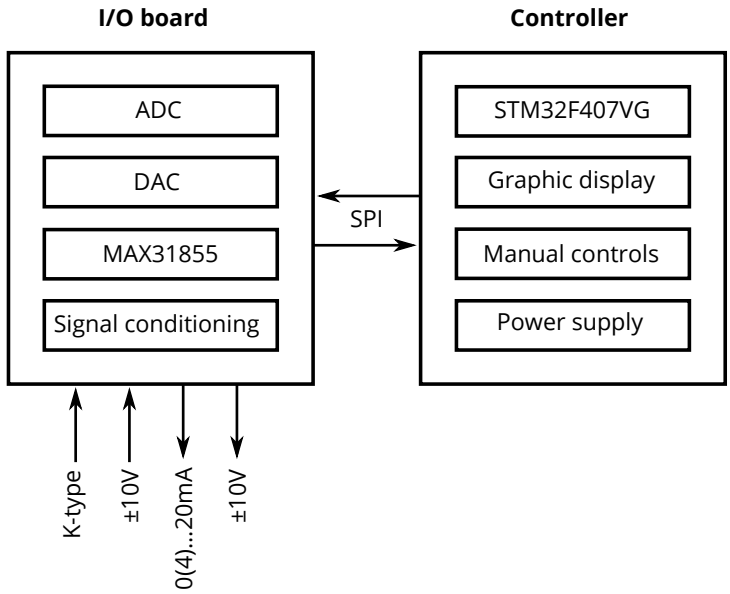


Figure 5.16: Structure schematic of the developed FOPID controller prototype based on STMicroelectronics STM32F407VG microcontroller

5.6 Conclusions

In this chapter, implementation methods for FO models and controllers were discussed, including both analog and digital realizations. An update to Carlson's method to FO capacitor approximation was proposed, which was found especially useful in generating FO lead-lag compensator type controller approximations. The method offers a sufficient rate of convergence and is based on a classical numerical method. A unified framework for analog realization of FO models and controllers was provided leveraging the object-oriented programming paradigm. The framework allows facilitating the choice of a particular network structure and approximation method for fractance circuit generation.

A digital implementation method based on the Oustaloup recursive filter approximation was proposed; the realization targets embedded device implementation. The proposed realization has several advantages. First, the filter implementation is always stable given that the corresponding floating-point computations are also stable. The chosen IIR filter structure allows to further tackle the problem of ensuring computational stability. A hardware implementation of a FOPID controller based on the previous results was presented and successfully verified in a number of real-time control experiments.

The most important issue encountered in the physical and computer-based realizations of fractional systems and controllers through use of approximations is the appropriate choice of either analog electrical component

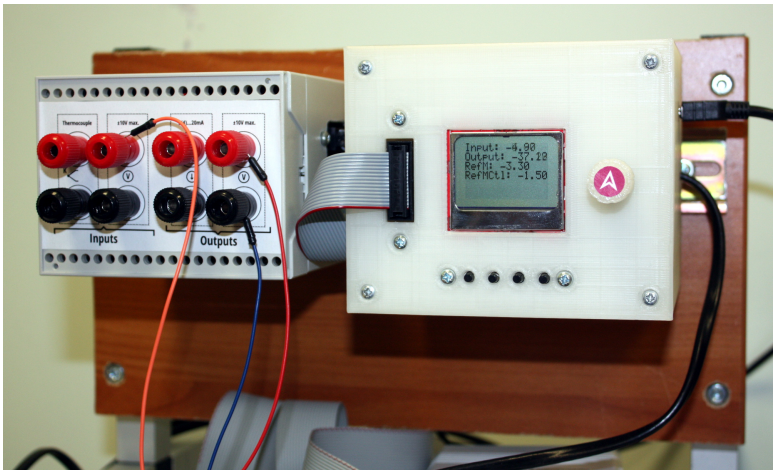


Figure 5.17: Assembled FOPID controller prototype and input/output board

values or coefficients of digital filters. In the former case, certain approximations may be unrealizable or infeasible due to high order or wide frequency band thereof since the obtained component values may become too large or too small. In case of digital filters, the difference in the magnitude of coefficient values may lead to loss of computational stability, especially in case of floating-point computations. One of the aims of this thesis was to facilitate the design of appropriate circuits and digital filters by providing means to tackle this issue. The obtained results indicate that this goal was practically achieved. However, there still exist certain limitations, e.g., while digital filter approximations of FOPID controllers run and maintain long-term stability on relatively low-end hardware, more complicated FO control algorithms necessitate the use of high-end hardware to run efficiently and perform correctly.

Some other issues were also found.

- The updated Carlson's method has certain limitations in that a considerable computational effort is needed to obtain the approximations, which, as a result, is of high order. This limits the use of the method compared to, e.g., Oustaloup's method.

- Optimization based analog realization of FO models and controllers following the discussion in Section 5.2 is not yet available. The proper formulation of the optimization problem in this case may allow obtaining approximations of complex fractance structures. It is expected, that global optimization algorithms will be very useful for solving this problem.

- The hardware prototype of the FOPID controller may benefit from the

use of the recently developed variable-order fractional operator methods [110].

The solution to these issues is the subject of future research.

Chapter 6

FOMCON: Fractional-order Modeling and Control Toolbox

In the following chapter, FOMCON toolbox for MATLAB/Simulink is described. The chapter has the following structure. First, an overview of the toolbox is provided in Section 6.1. Next, the identification, control, and implementation modules comprising the toolbox are described in Sections 6.2, 6.3, and 6.4, respectively. Illustrative examples are provided for the most important tools available in each module. Finally, in Section 6.5 conclusions are drawn.

6.1 Overview of the Toolbox

The FOMCON toolbox for MATLAB is a fractional-order calculus based toolbox for system modeling and control design. The core of the toolbox is derived from an existing toolbox FOTF ("Fractional-order Transfer Functions"), the source code for which is provided in literature. Consequently, the main object of analysis in FOMCON is a fractional-order transfer function of the form (2.11):

$$G(s) = \frac{b_m s^{\beta_m} + b_{m-1} s^{\beta_{m-1}} + \cdots + b_0 s^{\beta_0}}{a_n s^{\alpha_n} + a_{n-1} s^{\alpha_{n-1}} + \cdots + a_0 s^{\alpha_0}}. \tag{6.1}$$

FOMCON is related to other existing fractional-order calculus oriented MATLAB toolboxes, such as CRONE [90] and Ninteger [132] through either system model conversion features or shared code, and this relation is depicted in Figure 6.1.

The initial motivation for developing FOMCON was the desire to obtain a set of useful and convenient tools to facilitate the research of fractional-order systems. This involved writing convenience functions, e.g. the polynomial

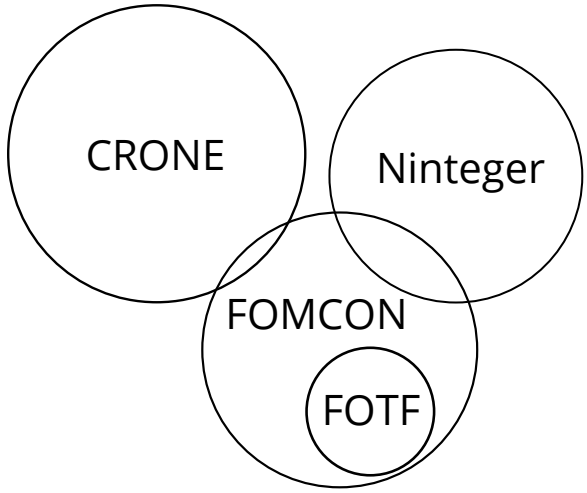


Figure 6.1: Relation of FOMCON toolbox for MATLAB/Simulink to similar packages

string parser, building graphical user interfaces to improve the general workflow. However, a full suite of tools was also desired due to certain limitations in existing toolboxes. The basic functionality of the toolbox was then extended with advanced features, such as fractional-order system identification and FOPID controller design.

With all previous considerations, the motivations for developing the toolbox can now be established.

- It is a product suitable for both beginners and more demanding users to to availability of graphical user interfaces and advanced functionality;
- It focuses on extending conventional control schemes (PID and lead-lag compensator loops) with concepts of fractional calculus;
- Tools for implementing fractional-order systems and controllers are available;
- With the Simulink blockset the toolbox aims at a more sophisticated modeling approach. Real-time control application support is provided through, e.g., Real-Time Windows Target toolbox for MATLAB/Simulink [65];
- It can be viewed as a "missing link" between CRONE and Ninteger;
- Due to availability of the source code the toolbox can be ported to other computational platforms such as Scilab or Octave (some limitations and/or restrictions may apply).

Most of the research results discussed in this thesis are implemented in FOMCON toolbox. Toolbox documentation is available on the official website [120].

Structure of the Toolbox

The toolbox has a modular structure depicted in Figure 6.2 and currently consists of the following modules:

- Main module (core—fractional system analysis);
- Identification module (system identification in both time and frequency domains);
- Control module (FOPID controller design, tuning and optimization tools, as well as some additional features);
- Implementation module (continuous and discrete time approximations, implementation of corresponding analog and digital filters).

All the modules are interconnected. Most features are supported by graphical user interfaces.

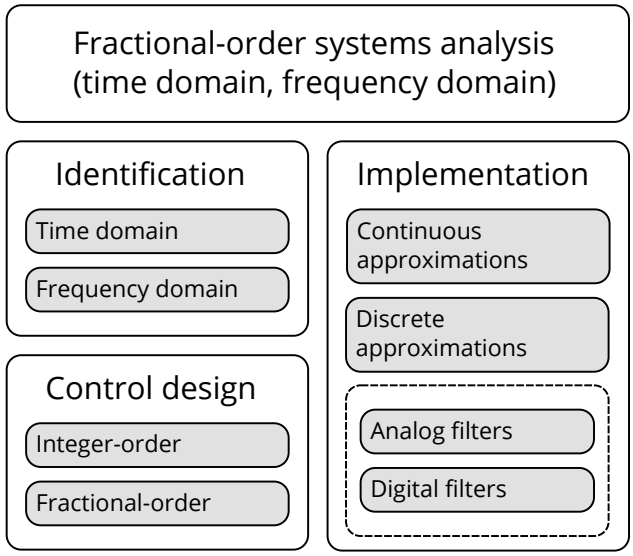


Figure 6.2: Modular structure of the FOMCON toolbox

A Simulink blockset is also provided in the toolbox allowing more complex modeling tasks to be carried out. General approach to block construction was used where applicable. The following blocks are currently realized:

- General fractional-order operators: fractional integrator and differentiator;

- Continuous and discrete time fractional transfer function;
- Continuous and discrete time FOPID controller.

Several variants of these blocks are provided for convenience.

Dependencies

The toolbox relies on the following MATLAB products:

- Control System toolbox—required for most features;
- Optimization toolbox—required for time domain identification and conventional PID tuning, and also partially for fractional-order PID tuning.

Several other tools are used directly (without or with minor changes) per the BSD license:

- Nelder-Mead algorithm based function for nonlinear optimization subject to bounds and constraints [84];
- Ninteger toolbox frequency domain identification functions [132].

It is also possible to export fractional-order systems to the CRONE toolbox format [90]. This feature requires the object-oriented CRONE toolbox to be installed.

6.2 Identification Module

The module provides the following main features:

- Time domain identification:
  - Commensurate and noncommensurate order system identification;
  - Parametric identification, which is applicable to closed-loop identification problems;
  - Approximation of fractional systems by conventional process models.
- Frequency domain identification:
  - Commensurate transfer function identification based on algorithms by Hartley, Levy and Vinagre [132];
  - Best fit algorithm for choosing an optimal commensurate order and pseudo-orders of the fractional transfer function [118].

In addition, functions for manipulating the obtained model are provided, including truncation, rounding and normalization of coefficients and orders, as well as functions for validating the models and carry out residual analysis.

Example 6.1 In this example we illustrate the use of the `fid` function of FOMCON toolbox. The task is to identify a system from an experimental signal to verify the identification algorithm. This is similar to Example 3.1, however, in this case the output of the system is not corrupted by noise. The system under study is described by

$$G(s) = \frac{-1.3333s^{0.63} + 2.6667}{1.3333s^{3.501} + 2.5333s^{2.42} + 1.7333s^{1.798} + 1.6667s^{1.31} + 1}. \quad (6.2)$$

An excitation signal—a PRBS7 sequence with amplitude of 1 applied for 30 seconds and immediately followed by a sine wave with an amplitude of 1 centered around zero with frequency of 20 Hz lasting 30 seconds—is applied to the input of this system and output samples are collected with a sample rate of 200 Hz. Assuming the variables y, u, and t hold the experimental output, input, and sample time vector, respectively, the initial model structure and parameters are chosen as

$$G_i(s) = \frac{s+1}{s^3 + s^{2.5} + s^{1.5} + s + 1}, \quad (6.3)$$

the coefficients are bounded such that $c \in [-100, 1000]$ and orders are bounded such that $q \in [10^{-9}, 5]$, and Oustaloup recursive filter approximations are used to simulate the model of the system with $\omega \in [0.0001, 10000]$, $N = 5$, the following code may be used to identify this system using the `fid` function and the Trust-Region-Reflective optimization algorithm:

```
% Setup: Create the fractional identification dataset
id1 = fidata(y, u, t);

% Initial model structure and parameters
g_i = fotf('s+1', 's^3+s^2.5+s^1.5+s+1');

% Use Oustaloup approximation for system simulation
fsp = fsparam(g_i, 'oust', [0.0001 10000], 5);

% Model is assumed to have a static gain
gp  = {1, []};

% Optimization algorithm: Trust-region-reflective
op.IdentificationAlgorithm = 'trr';
lim = {[-100; 1000], [1e-9 5]};     % Bounds

% Run the identification: G_id1 is the identified model
[~,~,~,~,~,G_id1] = fid(fsp, gp, id1, [], [], [], lim, op);
```

Table 6.1: Identification of a complex fractional system: results for different estimation algorithms

Algorithm	%Fit	ε_{MSE}	NoIter	FunEval	τ, min
TRR	99.98	$1.50 \cdot 10^{-8}$	84	1020	02:26
LM	99.07	$4.24 \cdot 10^{-5}$	97	1227	02:50

For invoking the Levenberg-Marquardt algorithm the following commands may be used:

```
% Optimization algorithm: Levenberg–Marquardt
op.IdentificationAlgorithm = 'lm'; op.Lambda = 100;

% Run the identification: G_id2 is the identified model
[~,~,~,~,~,G_id2] = fid(fsp, gp, id1, [], [], [], [], op);
```

A summary of the achieved results is provided in Table 6.1. In addition, frequency domain characteristics of obtained models are compared to those of the original model by means of a Bode diagram in Figure 6.3. The following models are obtained with the powers truncated:

$$G_{id1}(s) = \frac{-1.281s^{0.656} + 2.657}{1.396s^{3.495} + 2.145s^{2.471} + 2.736s^{1.817} + 1.199s^{1.176} + 1} \tag{6.4}$$

and

$$G_{id2}(s) = \frac{0.014s^{4.617} + 2.627}{0.899s^{4.922} + 5.003s^{3.409} + 6.519s^{2.059} + 1.71s^{0.962} + 1}. \tag{6.5}$$

Clearly, using the Trust-Region-Reflective estimation algorithm leads to a more accurate result in this particular case. While the time domain fit is very good in both cases, the comparison of frequency domain response clearly shows the difference in the models.

Example 6.2 This example shows the application of the parametric identification function `pfid` of FOMCON toolbox to the problem of closed-loop identification of fractional models using both the indirect and direct approaches from Section 3.3.

For the purpose of testing the identification algorithms for closed-loop systems, we consider the real-time prototyping platform discussed in Section 5.4. A linear fractional-order system, which is given by a nominal model of the form

$$G(s) = \frac{1}{0.8s^{2.2} + 0.5s^{0.9} + 1} \tag{6.6}$$

is running on the host computer in real-time Simulink software with the admissible control $u \in [0, 1]$, while a FOPID of the form

$$C(s) = 0.01 + \frac{0.53795s^{0.1}}{s} + 0.84749s^{0.75} \tag{6.7}$$

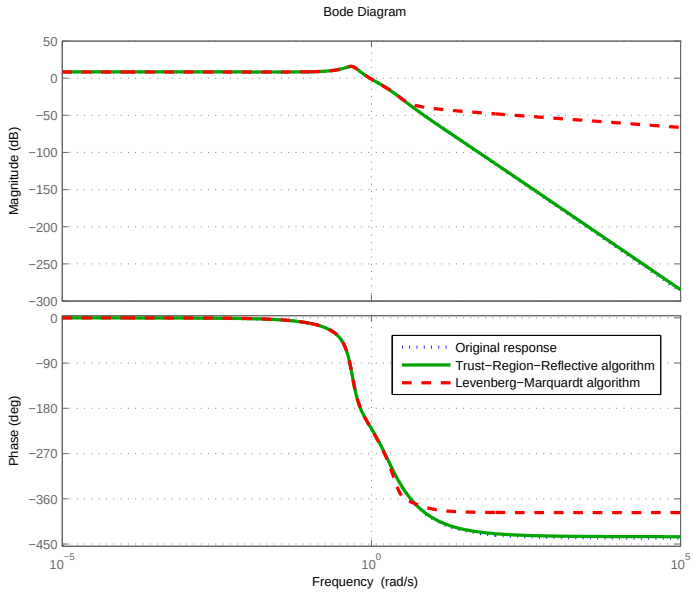


Figure 6.3: Identification of a complex fractional system: frequency domain response of the identified models and the original model

is implemented on the prototype from Section 5.5.1. Note that the fractional-order integrator is implemented as suggested in [72]. A constant reference signal $r(k) = 0.5$ is chosen, and experimental data forming the sets in (3.25) and (3.26) is collected. In the following, we supply this data to the identification algorithm with the aim of reconstructing the nominal transfer function in (6.6) thereby verifying the identification algorithm. Since we are dealing with a fractional-order system, available identification methods for linear, time-invariant systems, cannot be applied directly.

Suppose that the structure of a fractional-order model to be identified is known and may be parametrized as

$$G_p(s) = \frac{p_1}{p_2 s^{q_1} + p_3 s^{q_2} + p_4}. \tag{6.8}$$

The closed-loop transfer function in (2.32) used in the identification procedure is then given by

$$G_{cl}(s) = \frac{p_1 G_{pid}(s)}{s\left(p_2 s^{q_1} + p_3 s^{q_2} + p_4\right) + p_1 G_{pid}(s)}, \tag{6.9}$$

where $G_{pid}(s) = \left(K_p s + K_i s^{1-\lambda} + K_d s^{1+\mu}\right)$ and the parameters of the FOPID controller are assumed to be known and correspond to those in (6.7). Assuming that the `fid_ind` object holds the experimentally collected data, iden-

tification may be accomplished by means of the following MATLAB code, based on the discussion above:

```
% Linear controller parameters
m.Kp = 0.01; m.Ki = 0.53795; m.Kd = 0.84749;
m.Lambda = 0.9; m.Mu = 0.75;

% Closed-loop system
g_cl = ['(p1*(Kp*s+Ki*s^(1-Lambda)+Kd*s^(1+Mu)))/'...
        '(s*(p2*s^q1+p3*s^q2+p4)+' ...
        '(p1*(Kp*s+Ki*s^(1-Lambda)+Kd*s^(1+Mu))))'];

% Run the identification procedure
[prms, G] = pfid(fid_ind, g_cl, [], m);

% Restore the model
s   = fotf('s');
Gid = prms.p1 / ...
     (prms.p2*s^prms.q1+prms.p3*s^prms.q2+prms.p4);
```

The following model is recovered:

$$G_1(s) = \frac{0.980}{0.886s^{2.550} + 1.328s^{1.254} + 1.000}. \tag{6.10}$$

The results of system identification are presented in Figure 6.4. A frequency response of the original model is compared to that of the identified model in Figure 6.5. We can observe that there are noticeable differences between the two models. The achieved error norm is $\xi = 4.675 \cdot 10^{-2}$ and a fit of 95.35% to the experimental data. In this case, the critical factor is the simulated actuator saturation nonlinearity. Nevertheless, the model can still be used in controller design applications.

We now verify the direct approach. The model structure is once again assumed to be known and of the form (6.8). Assuming that the `fid_dir` object holds the experimentally collected data for this case, identification may be accomplished using

```
% Open-loop system
g_ol = ['p1/' ...
        '(p2*s^q1+p3*s^q2+p4)'];

% Run the identification procedure
[prms, G] = pfid(fid_dir, g_ol, [], m);
```

The model is recovered from the parameters in the same way as above. The resulting model is given by

$$G_2(s) = \frac{0.999}{0.779s^{2.218} + 0.465s^{0.916} + 1.000}, \tag{6.11}$$

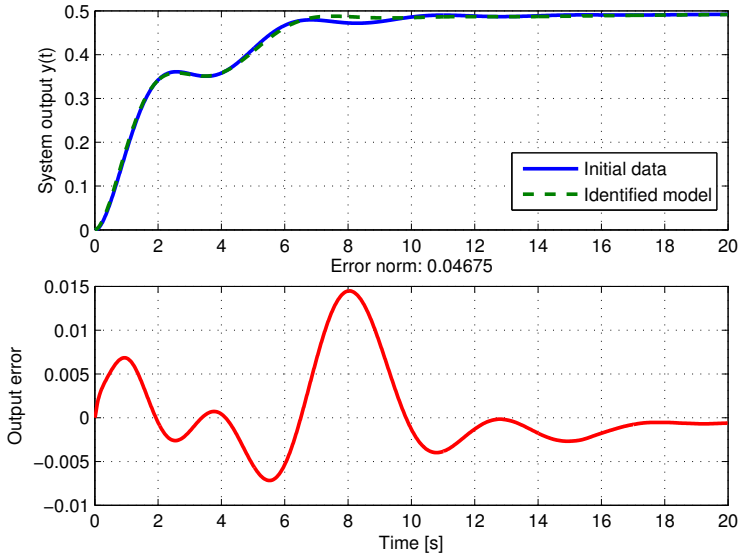


Figure 6.4: Indirect method: time domain validation

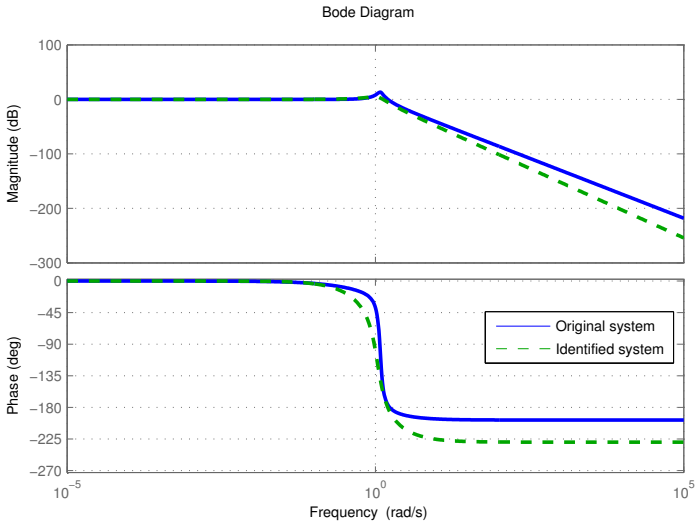


Figure 6.5: Indirect method: frequency domain validation

which is very close to the original nominal model with the error norm $\xi = 1.8374 \cdot 10^{-4}$ and a fit of 99.71% is achieved. The result has improved compared to the one obtained using the indirect approach. Slight discrepancies in the model parameters can be observed, however, they are in no way essential in terms of capturing the dynamics of the system under study. The results are also illustrated in Figures 6.6 and 6.7.

6.3 Control Module

The module provides the following main features:

- (FO)PID controller design:
  - Optimization based tuning following the discussion in Section 4.1, including support for controller design for linear and nonlinear plants;
  - Integer-order PID controller design by approximating fractional-order systems by conventional process models.
- Locating the stability boundaries for stabilizing controllers based on the method from Section 4.3;
- Implementation of retuning controllers discussed in Section 4.4 suitable for real-time experiments;
- Basic support for fractional lead-lag compensators and TID controllers [145].

In what follows, an example of tuning a FOPID controller for a process is provided.

Example 6.3 In this example we illustrate the use of the `fpid_optimize` function. Reconsider the plant in (6.2), where, in addition, there exists an actuator nonlinearity, such that admissible input values are in the range $u \in [-1, 1]$. The goal is to obtain a suboptimal FOPID controller for the control of this plant subject to the following specifications:

- Minimal gain margin $G_m = 10$dB, minimal phase margin $\varphi_m = 60°$;
- Robustness to gain variations property must be fulfilled in (4.8), i.e., the phase response of the open loop must be flat at a particular crossover frequency ω_c;
- The performance index considered is IAE in (4.3).

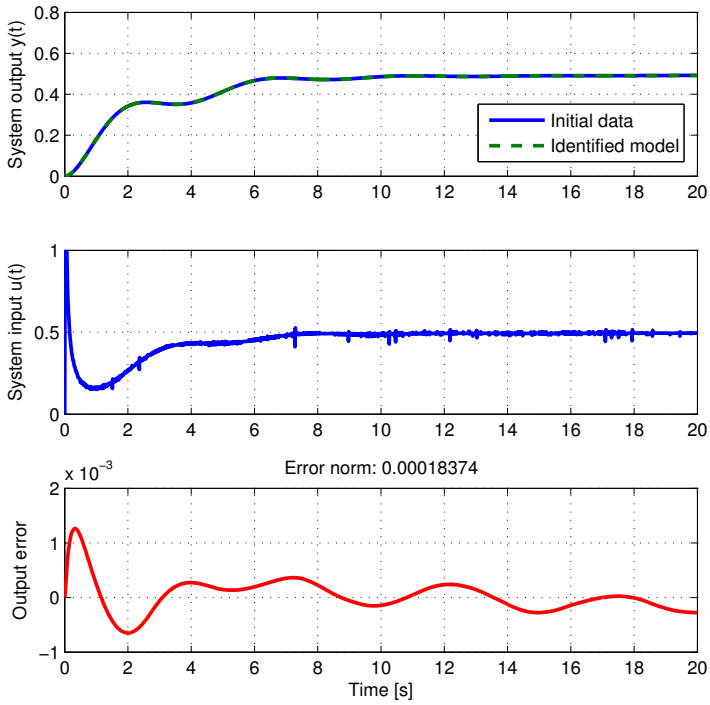


Figure 6.6: Direct method: time domain validation

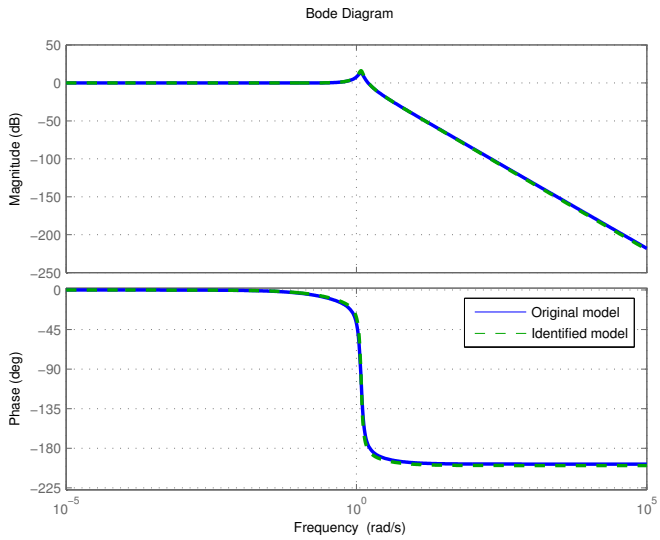


Figure 6.7: Direct method: frequency domain validation

For simulation of the control system, as well as for computation of frequency domain characteristics, the Oustaloup filter is used with the same parameters as in Example 6.1. The initial FOPID controller parameters are selected as

$$K_p = K_i = K_d = 1/K_{dc}, \quad \lambda = 0.9, \quad \mu = 0.7, \tag{6.12}$$

where $K_{dc} = 8/3$ is the static gain of the plant in (6.2). Optimization based tuning is carried out in two stages, each consisting of 20 Nelder-Mead algorithm iterations. This is done because if all specifications are imposed at the same time while the search variables are located too far from a solution, the use of penalty functions will lead to the saturation of the cost function, making further solution of the optimization problem difficult.

First, the gain and phase margin specifications are imposed, as well as the IAE performance metric. Then, a crossover frequency ω_c is recovered, and the robustness to gain variations requirement is imposed. After 20 more iterations of the optimizer, the suboptimal FOPID controller parameters are returned. The following MATLAB code may be used to achieve the set of tasks described above.

```
% Plant to be controlled
Gp  = fotf('-2s^{0.63}+4', ...
     '2s^{3.501}+3.8s^{2.42}+2.6s^{1.798}+2.5s^{1.31}+1.5');

% Use Oustaloup approximation for system simulation
fsp = fsparam(Gp, 'oust', [0.0001 10000], 5);

% Initial parameters
Kdc=dcgain(Gp); pin = [1/Kdc 1/Kdc 1/Kdc 0.9 0.7];

% Optimization options: load default values and set specifications
fpo = fpopt; fpo.p = pin; fpo.metric='iae';
fpo.pmax = [100 100 100 1 0.9]; fpo.pmin = [0 0 0 0.01 0.01];
fpo.margins=[10, 0; 60, 0]; fpo.ulim = [-1; 1]; fpo.sens = [];
fpo.gainvar = []; fpo.optop = optimset('maxiter', 20);

% Simulation time
fpo.simtime = [0.01 0.5 200];

%% First stage
[Kp, Ki, Kd, lam, mu] = fpid_optimize(fsp, fpo, ...
    [], 'fpid_optimize_model');

%% Second stage: get and use wcg
[~,~,~,wcg] = margin(fracpid(Kp, Ki, lam, Kd, mu)*Gp);
fpo.p = [Kp Ki Kd lam mu]; fpo.gainvar = [wcg; wcg];
[Kp, Ki, Kd, lam, mu] = fpid_optimize(fsp, fpo, ...
    [], 'fpid_optimize_model');
```

After the second stage, FOPID controller parameters are found as

$$K_p = 0.013989, \quad K_i = 0.045454, \quad K_d = 0.22067, \\ \lambda = 0.90128, \quad \mu = 0.68677. \quad (6.13)$$

Intermediate result comparison can be seen in Figures 6.8 through 6.11. Initial parameters result in an unstable control system, but in the process of optimization the control system is stabilized. In the second stage, the robustness to gain variations specification is fulfilled, which also results in improvement in the transient response. Thus, the given control task was successfully achieved.

6.4 Implementation Module

The module provides the following main features:

- Implementation of continuous-time approximation methods discussed throughout this thesis, e.g., Oustaloup approximation from Section 2.3 and the fractional zero-pole pair method proposed in Section 5.1;
- Implementation of the class proposed and described in Section 5.2 for fractance network generation in the form of analog filters;
- Discrete-time approximations are based on continuous to discrete time conversion methods, several convenience functions are available in FOMCON to facilitate this task, including a GUI for obtaining an approximation of a FOPID controller;

In what follows, two relevant examples are provided.

Example 6.4 In this example, our goal is to obtain an analog implementation a fractional controller for a model of a position servo

$$G(s) = \frac{1.4}{s(0.7s+1)} \mathrm{e}^{-0.05s} \quad (6.14)$$

identified in [72]. The design specifications are as follows: phase margin $\varphi = 80°$, gain crossover frequency $\omega_{cg} = 2.2$ rad/s. In the same reference paper, a controller design was proposed, based on robustness considerations derived from the desired frequency domain characteristics of the plant, in the form of a fractional lead compensator:

$$C(s) = \left(\frac{2.0161s+1}{0.0015s+1} \right)^{0.7020}. \quad (6.15)$$

We implement this controller using the method from Section 5.2. We choose $R_1 = 200k\Omega$ and $C_1 = 1\mu$F. The basic structure is the Foster II form

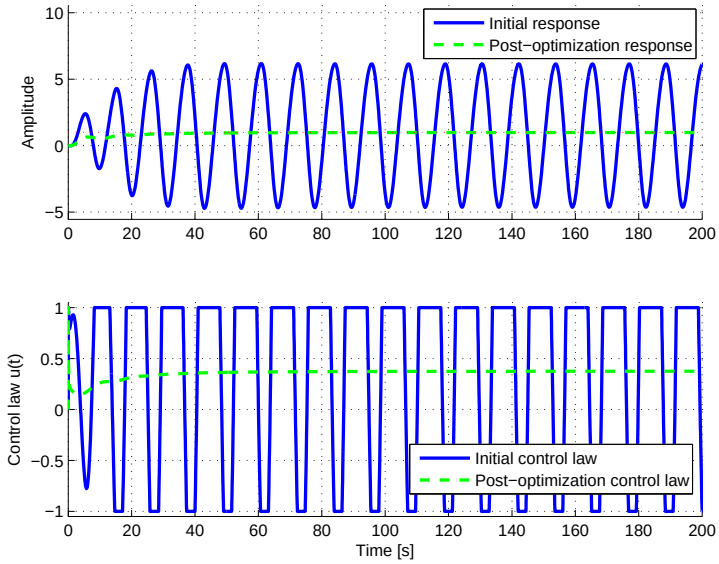


Figure 6.8: FOPID controller optimization example: from initial to suboptimal controller settings; time domain simulation

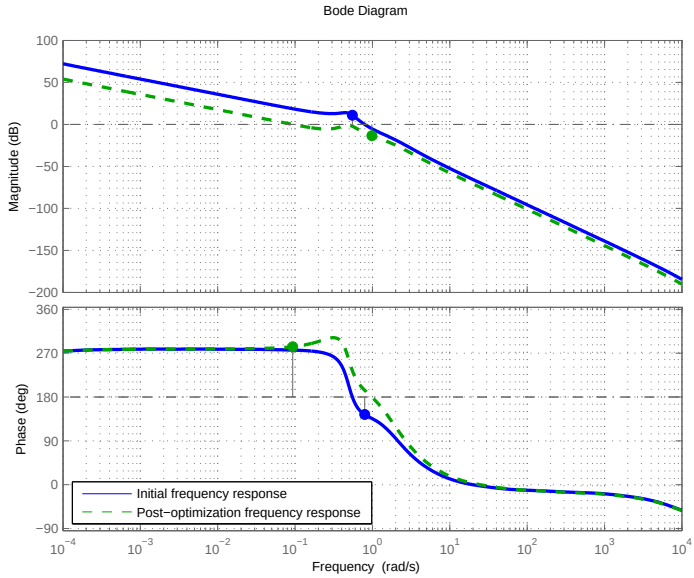


Figure 6.9: FOPID controller optimization example: from initial to suboptimal controller settings; frequency domain characteristics

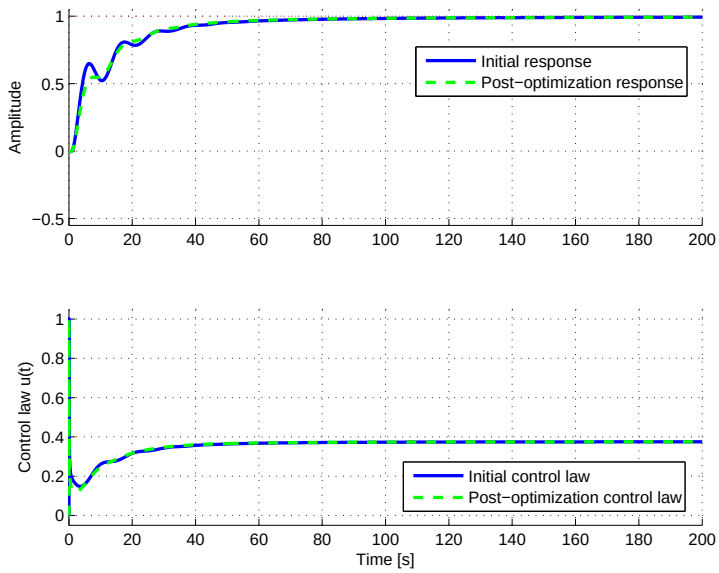


Figure 6.10: FOPID controller optimization example: robustness to gain variations specification fulfilled; time domain simulation

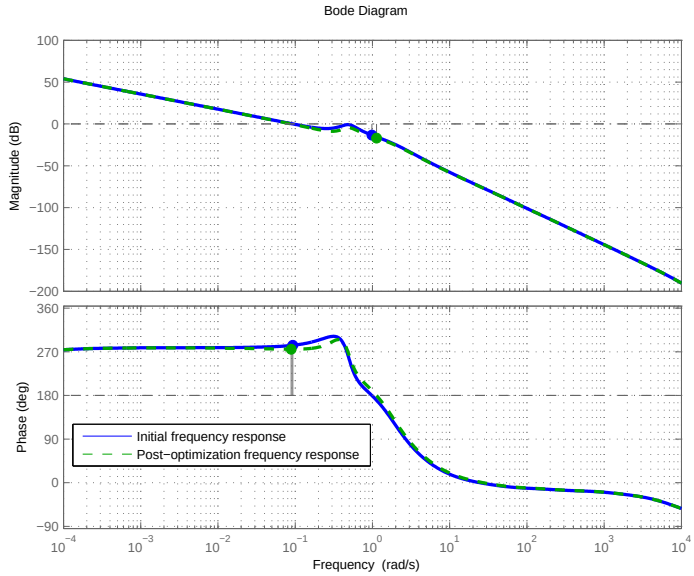


Figure 6.11: FOPID controller optimization example: robustness to gain variations specification fulfilled; frequency domain characteristics

RC network and the implementation is done by means of the described algorithm. To obtain the differentiator, we use the property $Z_d(s) = 1/Z_i(s)$, where $Z_d(s)$ and $Z_i(s)$ correspond to impedances of a differentiator and an integrator, respectively. This is done by setting the impedances in Figure 5.8 such that $Z_1(s) = Z_i(s)$ and $Z_2(s) = R_k$, where R_k serves as a gain correction resistor. To obtain the network approximation, the following MATLAB commands are executed:

```
% Parameters of the controller
b = 2.0161; wz = 1/b;
alpha = 0.702;
Gc = fotf('s')^alpha / wz^alpha;

% Specific fractance parameters
params = struct; params.R1 = 200e3;
params.C1 = 1e-6; params.N = 4;
params.varphi = 0.01;

% Obtain implementation
imp2 = frac_rcl(1/Gc, ...
       'frac_struct_rc_foster2', ...
       'frac_imp_rc_foster2_abgen', ...
       logspace(-2,3,1000), ...
       params);
```

Note that the transfer function approximation, corresponding to the controller, has to be obtained from the resulting circuit object by using

```
C = 1/zpk(imp2);
```

Following the successful generation of the network, the `prefnum` command is issued, which requests setting the resistor values to the preferred series with 5% tolerance, and the capacitor values substituted for closest components out of the 10%-series. This is done as follows:

```
imp2 = imp2.prefnum('5%','10%',[],'5%');
```

Finally, the user may wish to display the bill of materials by using the function `engnum`. For example, for the list of resistors one can use the command:

```
[vals, str] = engnum(imp2.R);
```

The variable `str` will contain string constants with the values of the resistors in the network:

```
'360 k'   '200 k'   '75 k'   '27 k'   '9.1 k'
```

The same could be applied to the gain setting resistor R_k, which in our case has the preferred value of $390k\Omega$. The resulting network is depicted in Figure 6.12. The schematic is simplified for simulation in LTSpice IV.

In Figure 6.13 a comparison of frequency responses of the ideal lead compensator, corresponding fractional-order differentiator and the resulting network approximation can be seen. As in the previous example, the frequency characteristics of the network were obtained from SPICE simulation and imported into MATLAB.

The open-loop frequency response of the control system is given in Figure 6.14. A shift in the design specifications can be observed. Corrections may need to be made to particular component values or the overall network structure must be enhanced. A real-life implementation was also considered, where the fractance network was physically assembled and tested. Several frequency response points were obtained to validate the realized network. The result of frequency domain identification can be seen in Figure 6.15. The resulting controller was also tested with the simulated plant using the prototyping platform from Section 5.4. The result of a real-time experiment versus a simulated experiment is presented in Figure 6.16. It can be seen that the resulting dynamics do not diverge in any significant way.

Example 6.5 In this example we obtain an IIR filter implementation of a PD$^\mu$ controller from [116]. The controller is given by

$$K_p = 0.055979, \quad K_d = 0.025189, \quad \mu = 0.88717. \tag{6.16}$$

The same method which is used in Section 5.3, that is, the zero-pole matching equivalents conversion is used. Finally, the second-order section form of the IIR filter is obtained in terms of coefficient arrays and the static gain in the form (5.49), which may be used directly in an algorithm developed in, e.g., C language. The following commands are executed in FOMCON:

```
% Obtain Oustaloup approximation of a PD^mu controller
Gc = zpk(oustapid(0.055979, 0, 0, 0.025189, 0.88717, ...
                  0.0001, 10000, 5, 'oust'));

% Convert to discrete-time with Ts = 0.01
Ts = 0.01;
Zc = c2d(Gc, Ts, 'matched');

% Display SOS arrays corresponding to
% the discrete-time approximation
d2sos(Zc);
```

and the following is displayed:

```
——— Coefficient arrays ———
{+1.0000000000, −0.9647855878, +0.0000000000},
```

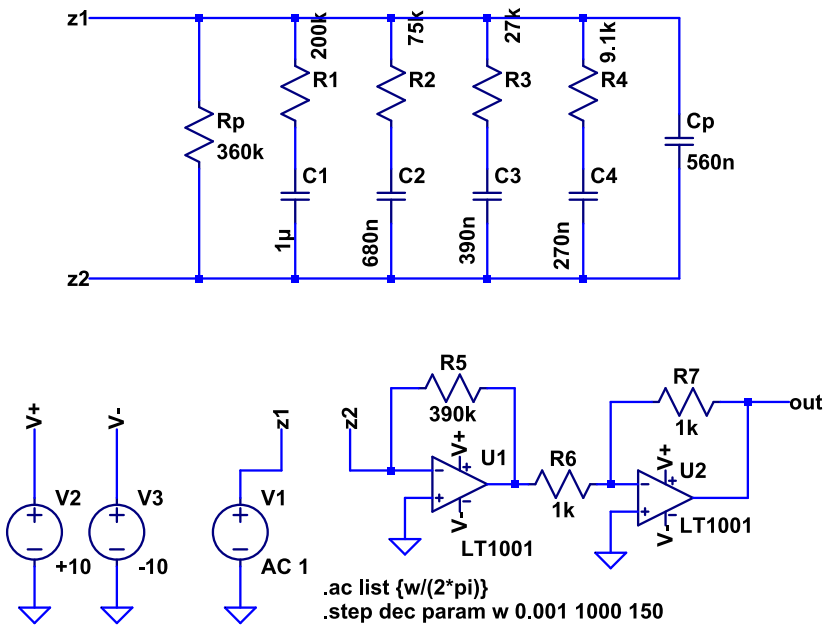


Figure 6.12: Fractional lead-lag compensator realization schematic in LTSpice IV

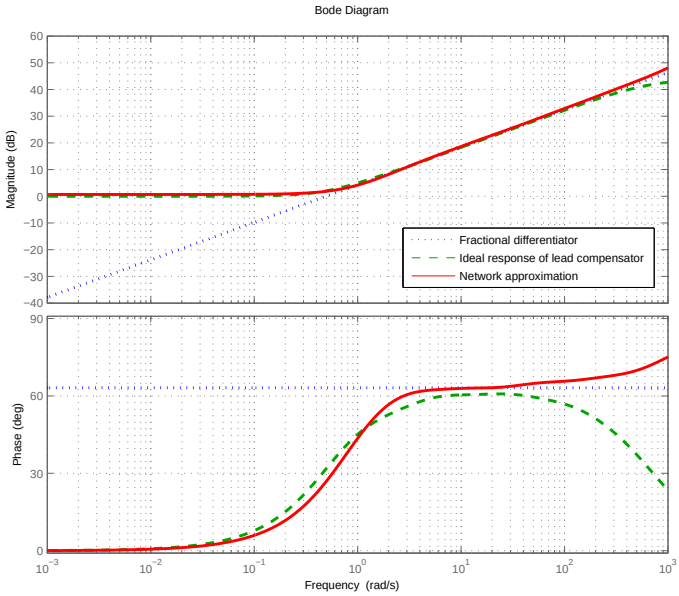


Figure 6.13: Frequency response of fractional differentiator $0.6113 \cdot s^{0.702}$ vs. lead compensator vs. network approximation implementation

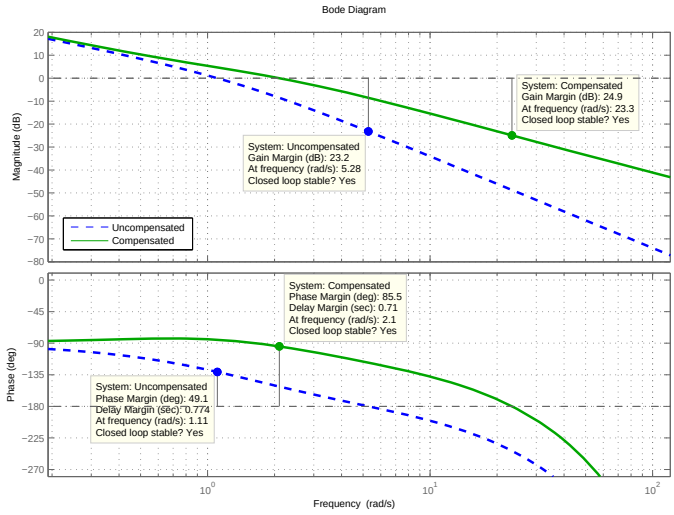


Figure 6.14: Open-loop frequency response of the controlled system

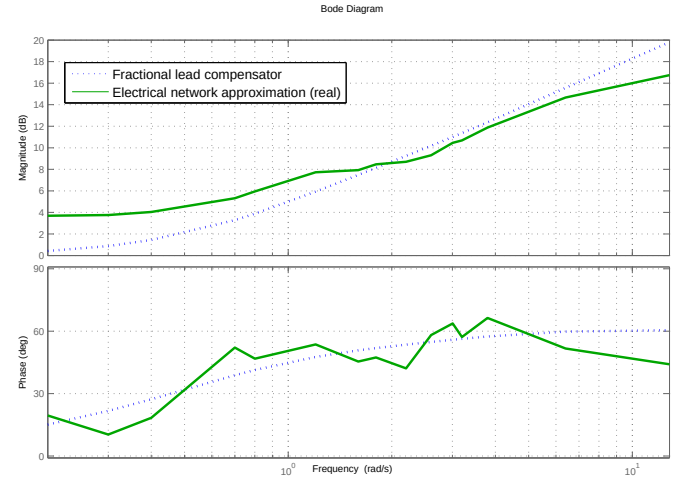


Figure 6.15: Real-life frequency response of the implemented fractance network

```
{+1.0000000000, −0.0209224276, +0.0000000000},
{+1.0000000000, −1.3493207288, +0.4180066451},
{+1.0000000000, −1.9807306143, +0.9807890156},
{+1.0000000000, −1.9991305017, +0.9991306026},
{+1.0000000000, −1.9999692428, +0.9999692429}

{+1.0000000000, −0.0000000000, +0.0000000000},
{+1.0000000000, −0.0409802515, +0.0000000016},
{+1.0000000000, −1.4434599048, +0.4912545169},
{+1.0000000000, −1.9752697983, +0.9753515564},
{+1.0000000000, −1.9991239831, +0.9991240851},
{+1.0000000000, −1.9999692318, +0.9999692319}


──── System DC gain ────
1.5336084022
```

The comparison of frequency response in case of the ideal PD$^{\mu}$ controller and the obtained approximation is given in Figure 6.17. The corresponding controller was implemented and successfully tested in [116].

6.5 Conclusions

In this chapter, FOMCON toolbox for MATLAB/Simulink was presented. The main focus was on MATLAB based features. The application of the tools available in the toolbox to solving identification, control, and analog and digital implementation problems for fractional systems was shown by means of a variety of examples conveying ideas from Chapters 3 through 5, including real-life application examples. The difference to closest known solutions [90, 132] is the focus on FOPID controllers and development of controller implementation methods for industrial applications.

The features of the toolbox completely cover the presented ideas, and its modular structure facilitates parallel development of these features and the object-oriented programming approach has additional benefits in this regard. Each major feature is accompanied with a graphical user interface thus making workflow more efficient.

The particular result is a complete software framework supporting FO model based control design and implementation of the obtained controllers.

It is expected that further development of the toolbox will lead to the following: Implementation of more specific identification algorithms, including those, that deal with identification of nonlinear fractional systems; Development of general, possibly global optimization based design algorithms for FOPID controllers, such that would also automate and facilitate the process of selection of particular design specifications; Realization and design

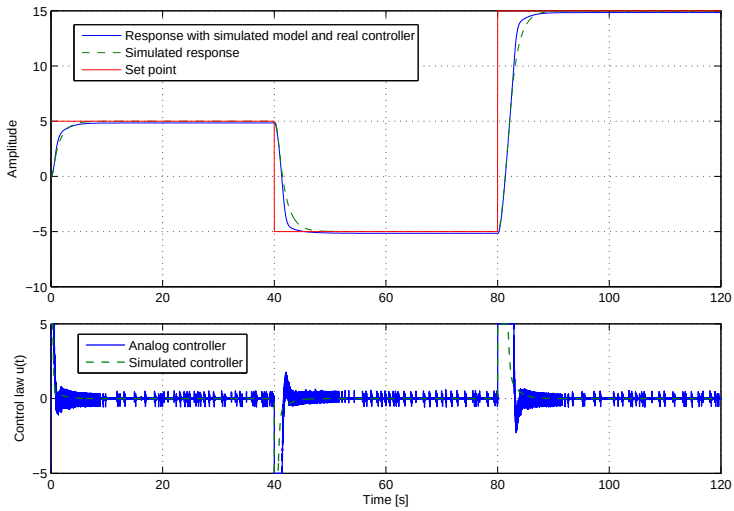


Figure 6.16: Real-life control experiment results with the implemented fractance network and a simulated plant

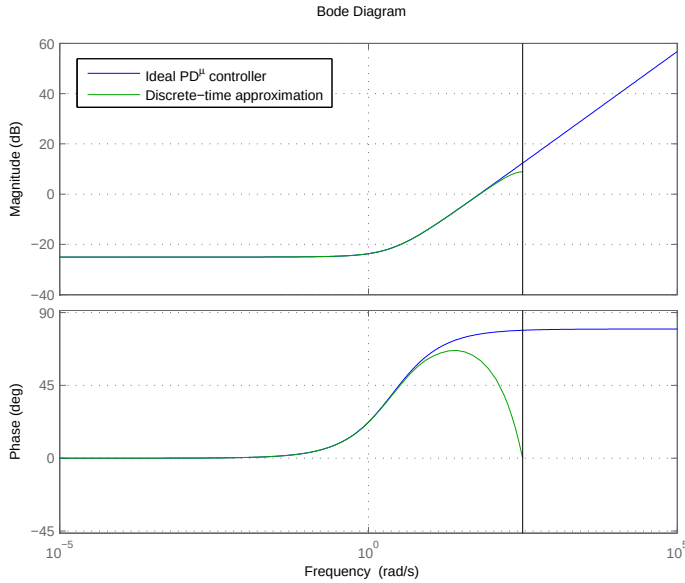


Figure 6.17: Frequency response of an ideal PD^{μ} controller vs. the obtained digital approximation

of more types of fractional controllers and control strategies (e.g., fractional sliding mode control); Development of analog filter design algorithms, based on fractance network approximations and global optimization algorithms.

In terms of development, MATLAB has been found to be a very suitable platform for the implementation of the toolbox. The most important reasons are as follows: MATLAB is popular and widely used in control system design; An abundance of high-level language features and functionality allows to significantly reduce the time it takes to implement and verify a particular algorithm; Real-time control experiments are possible by means of a specialized toolbox and the Simulink environment; Open source alternatives such as Octave [30] exist with compatible language syntax. This makes it possible to easily port the developed toolbox to allow the open source control community to benefit from it.

Regarding the features of the toolbox, other development directions are considered as well, such as identification and control of fractional-order MIMO systems. This, however, requires further research.

Chapter 7

Applications of Fractional-order Control

In this chapter, several real-life applications of fractional control are presented. The control objects used in this chapter include hardware laboratory models of industrial plants and an electromechanical actuator. The motivation for this chapter can be seen in the following. Suppose that model based control design is carried out using computer based simulations of dynamic systems of interest. Then, before the developed methods may be considered for control of real industrial plants, where a design mistake may result in production losses, it is important to evaluate the performance of the controllers using real-life models of the corresponding objects. The chapter has the following structure. In Section 7.1 the problem of level control in two different laboratory objects representing models of industrial multitank systems is investigated. In the latter case, the gain and order scheduling approach is used. In Section 7.2 a method for position control in a magnetic levitation system is presented—a controller design procedure comprising several steps is carried out. In Section 7.3 a particular ion-polymer metal composite actuator is studied and two types of controllers are designed for it. The advanced hardware prototype of the FOPID controller is used in a real-time experiment. Finally, conclusions are drawn in Section 7.4.

7.1 Fluid Level Control in a Multi Tank System

The multi tank systems considered in this section model a wide range of industrial processes [13], such as those found in chemical engineering [31, 33], food processing, and irrigation systems, as well as in nuclear power plants [37], and other branches of the industry, where the quality of the product depends on control loop performance. In these applications level control is mainly achieved by means of PID controllers, whereas in this work

we investigate the application of FOPID control. Two types of models are considered:

- The coupled tanks system comprises two interconnected tanks [41]. In case of this system we are concerned with controlling the level in the first tank, while the level in the second tank is considered a disturbance. An interesting observation can be made regarding the obtained linear fractional approximations of the process;
- The Multi-Tank system [47] comprises three tanks placed in a vertical configuration. We consider the problem of level control in the first two tanks (the top tank and the middle tank). The gain and order scheduling approach to FOPID controller design is applied to control of the level in the top tank.

In what follows, we describe the systems, provide the details pertaining to the controller design procedure, and discuss the obtained results.

7.1.1 Coupled Tanks System

In the continuous-time domain the model of the coupled tanks system considered in this work (Figure 7.1) can be described by the following differential equations

$$\begin{aligned} \dot{x}_1 &= \frac{1}{A}u_1 - d_{12} - w_1 C_1 \sqrt{x_1}, \\ \dot{x}_2 &= \frac{1}{A}u_2 + d_{12} - w_2 C_2 \sqrt{x_2}, \end{aligned} \tag{7.1}$$

where

$$d_{12} = w_{12} \cdot C_{12} \cdot \text{sign}(x_1 - x_2)\sqrt{|x_1 - x_2|}, \tag{7.2}$$

x_1 and x_2 are levels of fluid in the tanks, A is the cross section of both tanks, C_1, C_2, and C_{12} are flow coefficients, u_1 and u_2 are pump powers. The coupled tank system has three valves. Each of them can be either fully opened or fully closed. This is governed in the model by factors w_1, w_2, and w_{12} with $w_k \in \{0, 1\}$.

The laboratory plant used in this work is a coupled fluid tank system shown in Figure 7.2. The plant is connected to the PC via USB for data transfer. A Simulink library for real-time control of the plant is available.

The flow coefficients for this particular plant were identified experimentally and are as follows: $C_1 = 0.0292$, $C_2 = 0.0259$, and $C_{12} = 0.0267$. The cross-section A of both tanks is $10.18 \cdot 10^{-4}\text{m}^2$ and maximal pump power u_h for both pumps is $5.9174 \cdot 10^{-5}\text{m}^3/\text{s}$. The height of both tanks is 0.29 m.

Three types of experiments are considered:

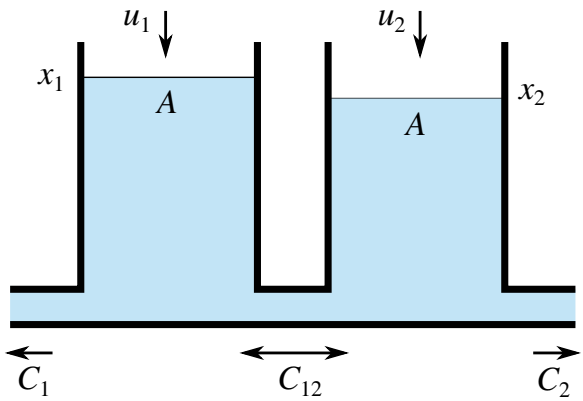


Figure 7.1: Coupled tanks system model

Figure 7.2: The coupled tanks system laboratory plant

1. Simulation of the fluid tank system model in MATLAB/Simulink used to optimize the controller parameters;
2. Control of the real plant from Simulink, the plant is connected to the computer via USB, no additional equipment is used;
3. Control of the sampled model in (7.1) running in Simulink in the configuration discussed in Section 5.4.

Note that the validation of the controller prototype is only done by using the plant model. The closed-loop behavior of the digital fractional PID controller prototype and the laboratory plant is yet to be investigated.

Single Tank Configuration

In this experiment, our goal is to control the level in a single tank, that is, the valve connecting the two tanks is closed, i.e. in the model (7.1) we have

$w_{12} = 0$. Next, we describe the procedure of tuning the PI$^\lambda$D$^\mu$ controller to control the real plant.

First, a fractional-order LTI model is identified from the step experiment. The identification data is recorded from the real plant and is transformed such that both input and output signals are in range $u, y \in [0, 1]$. The model obtained by means of the fractional system time domain identification tool of the FOMCON toolbox is as follows

$$G_1(s) = \frac{1}{5.3538 s^{1.0109} + 0.4126} \mathrm{e}^{-s}. \tag{7.3}$$

It can be seen that this model tends to an integer-order FOPDT model. The latter can be approximated using `iopid_tune` tool in FOMCON as

$$\hat{G}_1(s) = \frac{2.43567}{1 + 12.5914 s} \mathrm{e}^{-1.0779 s} \tag{7.4}$$

and integer-order PID controller gains can be obtained using the Ziegler-Nichols method as $K_p = 5.75515$, $K_i = 2.66962$, $K_d = 3.10175$.

The next step is to use the FOPID controller optimization tool of FOMCON toolbox using the obtained integer-order gains as initial values. The model (7.3) is approximated by a refined Oustaloup filter of order $N = 5$ within $\omega = [0.0001, 10000]$ rad/s. The chosen performance metric is integral absolute error (IAE). Simulink is used for system simulation and control effort saturation is considered. Gain and phase margins are specified as $G_m = 15$ dB and $\varphi_m = 90°$, respectively. Optimization set point is 0.5. Finally, the gains and orders of the controller are limited to the following ranges: $K_p = [0.1, 10], K_i = [0.001, 10], \lambda = 1, K_d = [0.1, 10], \mu = [0.5, 0.9]$. The selection of ranges is derived from considering the behavior of the fractional-order control actions in the time domain. After 100 iterations the following results are obtained:

$$K_p = 1.1434, \quad K_i = 0.079556, \quad K_d = 0.89839, \\ \lambda = 1, \quad \mu = 0.80468. \tag{7.5}$$

Upon implementing this discrete controller in MATLAB with sampling time $T_s = 0.1$s and testing the control loop in real time with the laboratory plant, these parameters resulted in a significant overshoot of the controlled level. After manually shifting the points $K_p' = 2K_p$ and $K_d' = 2K_d$ and further optimization, the following PID$^\mu$ controller was designed:

$$K_p = 2.6282, \ K_i = 0.097128, \ K_d = 1.26, \ \lambda = 1, \ \mu = 0.78894. \tag{7.6}$$

In Figure 7.3 the response of the control system in presence of random measurement errors caused by physical interference can be observed. It can

be seen, that the system easily recovers from such disturbance. The set point is $y = 0.2$m.

Next, we consider reference tracking. In this case, we also test the controller prototype using the obtained parameters. The set point changes from 0.2m to 0.15m at the 40th second of the simulation. The obtained results are given in Figure 7.4. It can be seen that the response of the control loop with the real plant is different from those, where a model was used. Therefore, the model should be revised accordingly.

The controller prototype implemented using the method described in Section 5.3 performs as expected. The corresponding control system response closely matches the simulated one. However, the control law produced by the physical controller suffers from limit cycles. This is because in case of the experiments with controller implementation for the coupled tanks system the sample resolution of both the DAQ board and the controller prototype was limited to 8 bits, thus reducing the dynamic range of control. This, in turn, results in quantization [52, 86]. It is possible to eliminate the oscillations by increasing the sample resolution.

Coupled Tanks Configuration

This time the task is to control the level of the first tank in a system of coupled tanks, when in (7.1) we have $w_1 = w_{12} = 1$, and the last valve w_2, which can open at any time, is seen as disturbance. In [41] a nonlinear control design approach was used, but in this work we use a linear PI$^\lambda$D$^\mu$ controller.

We obtain a linear fractional approximation of the real plant by means of the identification tool as in the previous case from a step experiment with $w_1 = w_{12} = 1, w_2 = 0$. The resulting fractional-order model is described by a transfer function

$$G_2 = \frac{1}{7.3986s^{0.9455} + 0.4095}\mathrm{e}^{-0.1s}. \tag{7.7}$$

We notice that this model no longer tends to exhibit integer-order dynamics as in case of (7.3).

Due to the value of the delay term the basic tuning formulae for integer-order PID tuning do not provide feasible results. However, it is possible to select some starting point manually and run optimization several times. However, it is important to choose the correct frequency domain specifications to ensure control system stability. In our case the goal is to minimize the impact of disturbance, so constraints on the sensitivity functions could be imposed. Our choice is such that $|T(j\omega)| \leq -20\,dB, \forall\omega \geq 10\,\mathrm{rad/s}$ and $|S(j\omega)| \leq -20\,dB, \forall\omega \leq 0.1\,\mathrm{rad/s}$. The gain and phase margins are set to $G_m = 10\,\mathrm{dB}$ and $\varphi = 60^\circ$, respectively. Additionally, due to the same problem as in the previous experiment, in order to limit the overshoot, the upper

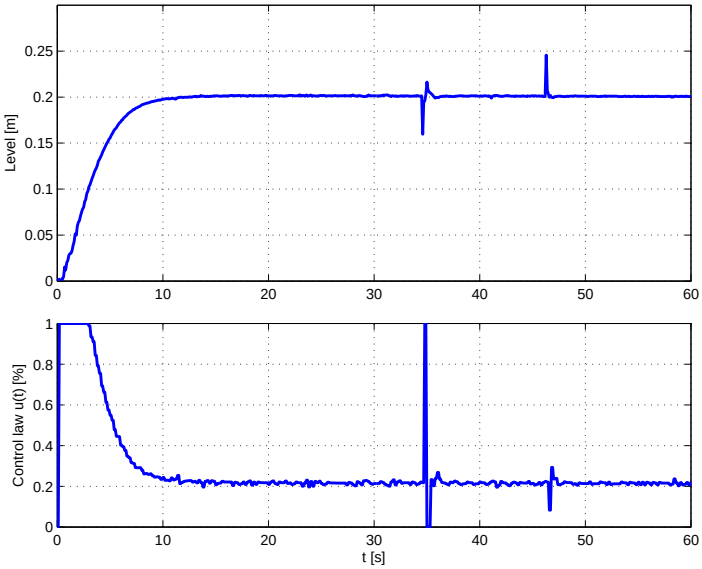


Figure 7.3: Level control with measurement noise

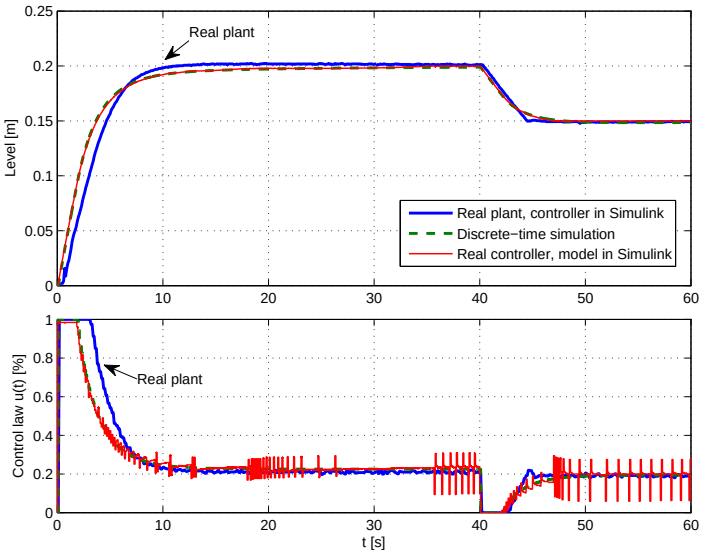


Figure 7.4: Level control with set point change

bound of control signal saturation was lowered from 100% to 60%. Using the IAE performance metric we finally arrive at the following PI$^\lambda$D$^\mu$ controller parameters by optimizing the response of the nonlinear system in Simulink:

$$K_p = 6.9514, \quad K_i = 0.13522, \quad K_d = -0.99874, \\ \lambda = 0.93187, \quad \mu = 0.29915. \quad (7.8)$$

Next, the discrete approximation is obtained. Note that in case of this controller it is very important to choose the proper frequency range of approximation. The suitable range for a refined Oustaloup filter of order 5 was found such that $\omega = [0.0001, 100]$ rad/s. The sampling time is $T_s = 0.2$s. The responses of the three control systems in different configurations are depicted in Figure 7.5. The set point is once again fixed at $y = 0.2\,\mathrm{m}$. Additional disturbance is such that the second pump is switched on at full capacity at $t = 76.4\,\mathrm{s}$ and is turned off at $t = 80.6\,\mathrm{s}$. Once again we observe that the behavior of the real plant differs from the simulated one while the responses of the discrete controller in Simulink and the digital controller are virtually identical. The control law oscillations of the digital controller are of smaller amplitude than in the previous experiment with the single tank. The obtained control system is capable of maintaining the set level with a tolerance of 5% in the presence of disturbances discussed above.

7.1.2 Multi-Tank System

First, we describe the particular configuration of the multi-tank system [47] and state the control problem.

The multi-tank system consists of three distinctly shaped tanks with level sensors and mechanical and automatic valves, a water reservoir, and a pump, that connects the reservoir and the upper tank. In this work we consider level control in the first two tanks, that is, in the upper one and the middle one. An illustration is provided in Figure 7.6.

This system can be described by the following differential equations:

$$\begin{aligned} \dot{x}_1 &= \frac{1}{\eta_1(x_1)} \left(u_p(v) - C_1 x_1^{\alpha_1} - \zeta_1(v_1) x_1^{\alpha_{v1}} \right), \\ \dot{x}_2 &= \frac{1}{\eta_2(x_2)} \left(q + r - C_2 x_2^{\alpha_2} - \zeta_2(v_2) x_2^{\alpha_{v2}} \right), \end{aligned} \quad (7.9)$$

where x_1 and x_2 are levels in the upper tank and middle tank, respectively, $\eta_1(x_1) = A = aw$ is the constant cross-sectional area of the upper tank, $\eta_2(x_2) = cw + x_2 bw / x_{2max}$ is the variable cross-sectional area of the middle tank, $u_p(v)$ is the pump capacity that depends on the normalized control input $v(t) \in [0, 1]$, $\zeta_1(v_1)$ and $\zeta_2(v_2)$ are variable flow coefficients of the automatic valves controlled by normalized input signals $v_1(t), v_2(t) \in [0, 1]$,

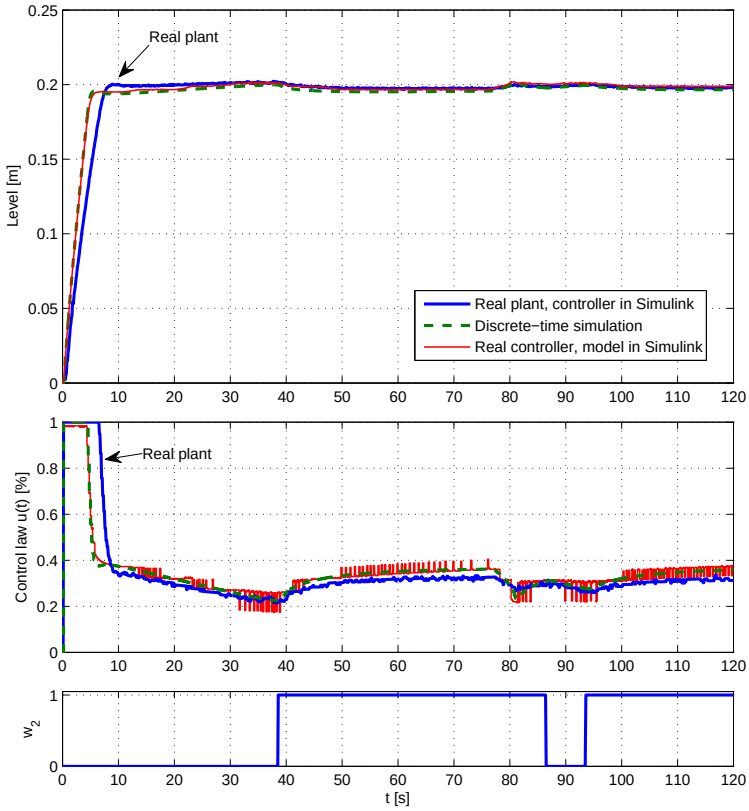


Figure 7.5: Level control in a system of coupled tanks

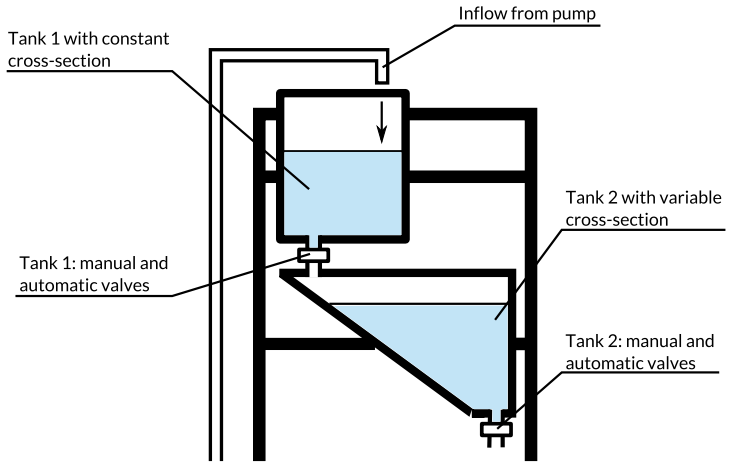


Figure 7.6: Configuration of the multi-tank system

$q = C_1 x_1^{\alpha_1}$ and $r = \zeta_1(v_1) x_1^{\alpha_{v1}}$. Fluid levels are taken as outputs of the system, i.e. $y_1 = x_1$, $y_2 = x_2$. A list of parameters of the model with their respective physical meaning is provided in Table 7.1. Note that for this system we have chosen to use a different approach to modeling the nonlinearity in the level dynamics due to pressure than in case of the coupled tanks system. The reason is that this model is more complicated due to the addition of the automatic valves, and to improve modeling accuracy, the states x_1 and x_2 in (7.9) are raised to powers α_1 and α_2.

Table 7.1: Parameters of the two-tank system

Parameter	Physical description
w	width of both tanks
a	length of the upper tank
b, c	lengths of the top and bottom part of the middle tank
x_{2max}	maximum attainable fluid level in the middle tank
C_i	resistance of the output orifice of the ith tank
α_i, α_{vi}	state exponents describing the flow of the ith valve

The multi-tank system may be controlled by means of a personal computer running MATLAB and Simulink software packages. The identification of parameters above is done by means of a series of experiments using nonlinear model estimation technique based on the Nelder-Mead optimization algorithm. The validation of the model in (7.9) was carried out under a variety of excitation signals. It can be seen from Figure 7.7 that a sufficiently

accurate model is obtained.

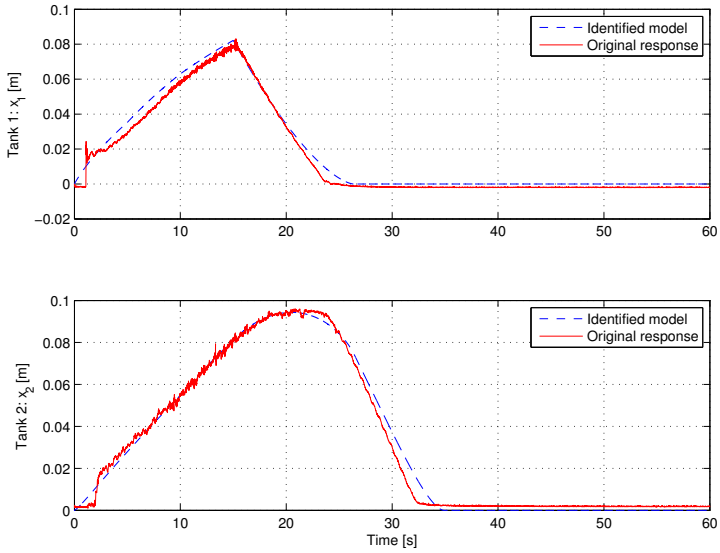


Figure 7.7: Model validation results

The control task is to design a controller for the upper tank such that would keep the level of fluid within reasonable bounds at the desired set point in the presence of disturbances caused by the controlled output valve. Also it is required to design a controller for the middle tank, such that would keep the level of fluid at the desired set point using controlled valves of the upper tank and also its own valve. To this end, we will now define a unified control input for controlling the level in the second tank $v_c(t) \in [-1, 1]$ such that the control inputs of the automatic valves are given by the following set of rules

$$v_1 = \begin{cases} 0, & \text{if } v_c \leqslant 0, \\ 0.3v_c + v_d, & \text{if } v_c > 0, \end{cases} \tag{7.10}$$

and

$$v_2 = \begin{cases} 0, & \text{if } v_c \geqslant 0, \\ -0.3v_c + v_d, & \text{if } v_c < 0. \end{cases} \tag{7.11}$$

The value $v_d = 0.7$ corresponds to the deadzone of the control in both cases, that is, the fluid does not flow through the automatic valves when $v_1 \leqslant v_d$ or $v_2 \leqslant v_d$. The constructed control law allows to regulate the fluid level in the middle tank. The tanks are, in fact, coupled, in the sense that only a limited range of fluid level values is achievable in the middle tank and it is related to the level in the upper tank. The outflow of liquid from the upper tank through the automatic valve forms part of the control for the middle

tank and is considered a disturbance from the perspective of level control in the upper tank.

One significant problem with the current implementation of the multi-tank system is the amount of noise present in level measurements. Since the ends of the submerged sensor tubes are placed very close to the output valves in all three tanks, the switching of the automatic valves creates additional noise which cannot be easily dealt with using linear filtering methods alone. Therefore, it was decided to employ an extended Kalman filter [107] to tackle this problem. Details may be found in [121].

Experimental Results

We now illustrate the use of the method proposed in Section 4.2 to the problem of level control in the two tanks. Numerical analysis is done by means of the FOMCON toolbox for MATLAB. The real-life system used in the experiments is depicted in Figure 7.8.

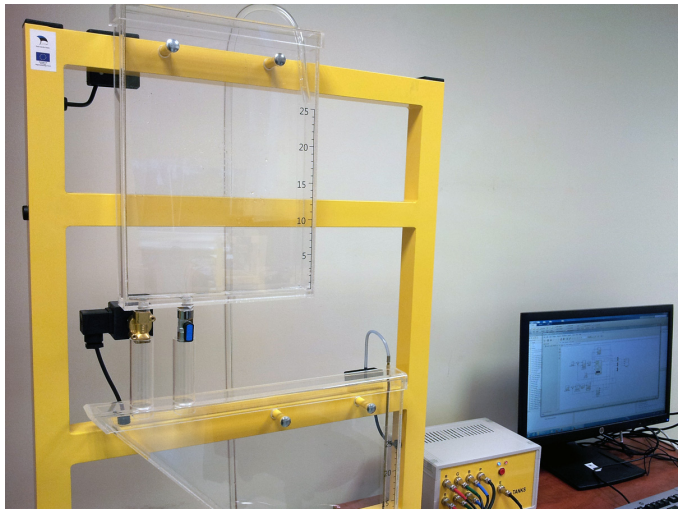


Figure 7.8: Laboratory model of a multi-tank system

First, linear approximations of the form (2.14) are obtained from the nonlinear model by means of time domain identification at system working points $(0.7029, 0.1)$ and $(0.7879, 0.2)$. The identification tool from Section (6.2) is used. The following models are found:

$$G_1(s) = \frac{0.14464}{18.728s^{0.91746} + 1} \tag{7.12}$$

and

$$G_2(s) = \frac{0.25881}{27.859s^{0.9115} + 1}. \tag{7.13}$$

During our study we have found that the fractional exponent α in the models above frequently arises during the identification process, and more importantly, these fractional models offer superior precision than classical ones in terms of describing the transient. With variation of certain model parameters we have identified a range that the exponent α can take to be approximately $\alpha \in (0.88, 0.94)$. This result may, in fact, be related to the one, appearing in case of the coupled tanks system. However, no clear relation has been identified between the fractional exponents of the process models and those in the states of the nonlinear model in (7.9).

Next, controllers are designed for level control in the upper tank using the FOPID optimization tool of FOMCON toolbox described in Section (6.3). For this a nonlinear model of the system is used for simulations in the time domain, the set value corresponds to the particular operating point. Linear fractional-order approximations corresponding to the working points are used to constrain the optimization process by employing frequency domain specifications.

In particular, in case of the first controller, a phase margin is set to $\varphi_m \geqslant 60^\circ$, sensitivity and complementary sensitivity function constraints are set such that $\omega_t = 0.02$ and $\omega_s = 0.1$ with $A = B = -20$ dB. Robustness to gain variations specification is also used with the critical frequency $\omega_c = 0.1$. For the second controller, the phase margin specification is changed to $\varphi_m = 85^\circ$ and the bandwidth limitation specified by ω_c is removed. Due to the flexibility of the tuning tool, it is possible to retune the controllers by considering the composite control law in (4.19) during the controller optimization process.

Finally, two FOPID controllers are obtained:

$$C_1(s) = 6.1467 + \frac{1.0712}{s^{0.9528}} + 0.8497 s^{0.8936}$$

and

$$C_2(s) = 5.1524 + \frac{0.3227}{s^{1.0554}} + 2.4827 s^{0.010722}.$$

The composite control law

$$C(s) = \frac{(1 - \gamma(x_1))\, C_1(s) + \gamma(x_1) C_2(s)}{2}$$

is then verified with both models $G_1(s)$ and $G_2(s)$ using (4.23) with step size of $\gamma(x_1) = 0.01$ and minimum commensurate order $q_{min} = 0.01$. The result of the test is that the closed-loop systems are stable in case of both fractional models.

Once the gain and order scheduled composite controller is designed, it is plugged into the control system, and a FOPID controller is designed for the control of the level in the second tank using the same optimization tool of the

FOMCON toolbox. Here we need to consider the following. First, frequency domain specifications are not applicable, since we do not have a linear model of this process. In addition, the application of the D^{μ} component is not very desirable in this case, because of the relatively large dead-zone in control of the outflow and therefore any noise amplified due to the differentiator component will lead to rapid switching of the automatic valves.

Therefore we design a FOPI controller based only on optimization of the transient response of the control system in the time domain. The following controller is obtained:

$$C_3(s) = 5.0000 + \frac{0.06081}{s^{0.1029}}$$

which is essentially a proportional controller with a weak fractional-order integrator.

Finally, the performance of the whole control system is evaluated with the real-life plant. All controllers are implemented by means of the Oustaloup filter approximation from Section 2.3. A low-pass filter is added to the level control loop of the second tank. This introduces a phase lag that reduces switching of the automatic valves. Set-points changes are presented in Table 7.2. The result of the performance evaluation is presented in Figure 7.9.

Table 7.2: Reference signal changes considered for the real-time control experiment with the Multi-Tank system

Time instance [s]	Upper tank reference [m]	Middle tank reference [m]
0	0.1	0.1
100	0.15	0.1
150	0.15	0.2
250	0.2	0.2

The levels in the first and second tanks are kept within reasonable error bounds of approximately 5% of the set point. The control task put forth previously is thereby accomplished. Notice that the noise does not propagate into the control law of the pump. The control law of the valves is, for the most part, bang-bang due to the large deadzone in the control of the automatic valves. However, the switching frequency is acceptable.

7.2 Retuning Control of a Magnetic Levitation System

The Magnetic Levitation System (MLS) is a nonlinear, open-loop unstable, and time-varying system [47]. Therefore, designing a stabilizing controller

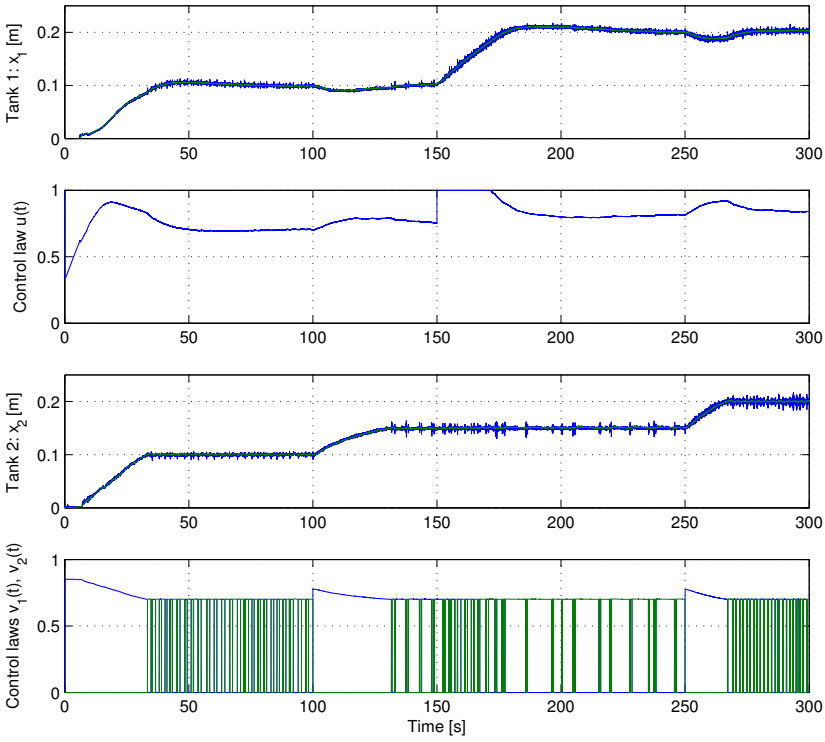


Figure 7.9: Control system evaluation results with the real-life plant

for it is a challenging problem. Yet it is also of significant importance, since MLS has a considerable range of real-life applications—it is used in, e.g., high-speed magnetic levitation passenger trains and vibration isolation of sensitive machinery [39]. Corresponding nonlinear control design methods were proposed in, e.g., [2, 9, 106]. However, few research papers deal with control design for unstable systems [77], and, in particular, for the MLS, which forms the motivation for our present research effort. Since FOPID controllers offer more possibilities to stabilize a plant under control than their conventional counterparts, the use of the former is favorable in case of control of the MLS.

Model of the MLS

The MLS considered in this work consists of an electromagnet, a light source and sensor to measure the position of the levitated sphere, and a sphere rest, the height of which is variable and determines the initial position x_{max} of the sphere in control experiments. The position of the sphere is determined relative to the electromagnet and has an effective range of $x \in [0, x_{max}]$ mm. A schematic drawing depicting this configuration is given in Figure 7.10. The basic principle of MLS operation is to apply voltage to the electromagnet to keep the sphere levitated [47].

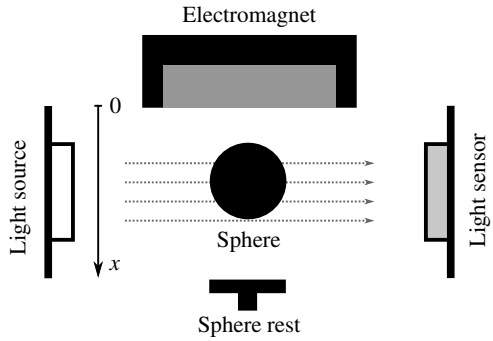


Figure 7.10: Physical model of the MLS

In [39] and [77] the following dynamical model for the MLS is used:

$$m\ddot{x} = mg - \frac{ci^2(u)}{x^2}, \tag{7.14}$$

where m is the mass of the sphere, x is the position of the sphere, g is gravitational acceleration, $i(u)$ is a function of voltage corresponding to the electrical current running through the coil of the electromagnet under input u, and c is some constant. However, the following practical observations can be made:

- It is essential to model the dynamics of the electrical current running through the coil;
- The parameter c is, in fact, not constant and depends on the position of the sphere x.

Therefore, we use the model description provided by INTECO, which takes into account the dynamics of the coil current. In addition, we model the parameter c by a polynomial $c(x)$. The following model is finally established:

$$\begin{aligned} \dot{x}_1 &= x_2, \\ \dot{x}_2 &= -\frac{c(x_1)}{m}\frac{x_3^2}{x_1^2} + g, \\ \dot{x}_3 &= \frac{f_{ip2}}{f_{ip1}}\frac{i(u) - x_3}{\mathrm{e}^{-x_1/f_{ip2}}}, \end{aligned} \tag{7.15}$$

where x_1 is the position of the sphere, x_2 is the velocity of the sphere, and x_3 is the coil current, f_{ip1} and f_{ip2} are constants. By means of a series of experiments we have found that it is sufficient to model $c(x_1)$ as a 4th order polynomial of the form

$$c(x_1) = c_4 x_1^4 + c_3 x_1^3 + c_2 x_1^2 + c_1 x_1 + c_0, \tag{7.16}$$

and $i(u)$ as a 2nd order polynomial of the form

$$i(u) = k_2 u^2 + k_1 u + k_0. \tag{7.17}$$

Note that the voltage control signal is normalized and has the range $u \in [0, 1]$ corresponding to the pulse-width modulation duty cycle $0 \ldots 100\%$.

We will analyze the stability of linear approximation around a working point (u_0, x_{10}). We linearize the model in (7.15) and obtain the following transfer function of the MLS:

$$G_M(s) = \frac{b_3 a_{23}}{s^3 - a_{33}s^2 - a_{21}s + a_{21}a_{33}}, \tag{7.18}$$

where

$$a_{21} = \frac{(-2c_4 x_{10}^4 - c_3 x_{10}^3 + c_1 x_{10} + 2c_0)x_{30}^2}{m x_{10}^3}, \tag{7.19}$$

$$a_{23} = -\frac{2c(x_{10})x_{30}}{m x_{10}^2}, \quad a_{33} = \frac{i(u_0) - x_{30}}{f_{ip1}}\mathrm{e}^{x_{10}/f_{ip2}}, \tag{7.20}$$

$$b_3 = \frac{f_{ip2}}{f_{ip1}}(k_1 + 2k_2 u_0)\mathrm{e}^{x_{10}/f_{ip2}}. \tag{7.21}$$

Summary of the Control Design Method

In the following, we summarize the method that shall be used to design FOPID controllers for the MLS.

1. Choose an operating point (u_0, x_{10}) and obtain a linear approximation of the form (7.18);
2. Use the method from Section 4.3 to locate a stabilizing FOPID controller;
3. Optimize the resulting FOPID controller using the feasible parameter ranges obtained in the previous step.

It poses great difficulty to impose feasible robustness specifications in case of optimization based FOPID controller design for the MLS. Thus, suboptimal controllers may be designed. The performance of the system will be evaluated experimentally, settling time τ_s, percent overshoot θ, and percent maximum deviation from reference due to disturbance θ_d are used as performance measures. In essence, we consider time domain simulations of the nonlinear model in (7.15) and minimize a cost defined by the ISE performance metric in (4.2). The choice of this particular performance index is dictated by the necessity to minimize the overshoot [39]. The optimization procedure is carried out by means of the method described in Section 4.1.

To analyze the stability of the closed-loop fractional-order control system in (2.32) we shall use Matignon's theorem. The characteristic polynomial is given by

$$\begin{aligned} Q(s) &= s^{3+\lambda} - a_{33}s^{2+\lambda} - a_{21}s + (b_3a_{23}K_p + a_{21}a_{33})s^{\lambda} \\ &\quad + b_3a_{23}K_ds^{\lambda+\mu} + b_3a_{23}K_i. \end{aligned} \tag{7.22}$$

Thus, a point of the form $(K_p, K_i, K_d, \lambda, \mu)$ in the PI$^{\lambda}$D$^{\mu}$ parameter space can be selected and the stability of the closed-loop control system can be verified.

For the purpose of validating our control design approach we use a real-life MLS provided by INTECO [47] and depicted in Figure 7.11. It is connected to a computer running MATLAB/Simulink thereby allowing to conduct real-time experiments. The specific parameters of the model in (7.15) are as follows: $m = 0.0585$ kg, $x_{max} = 0.0155$ m, $g = 9.81$ m/s^2. Other parameters need to be identified. The corresponding procedure is detailed in the following subsection.

7.2.1 Identification of the Nonlinear Model of the MLS

Our task is to identify two functions $i(u)$ and $c(x)$, as well as parameters f_{ip1} and f_{ip2} of the nonlinear model in (7.15).

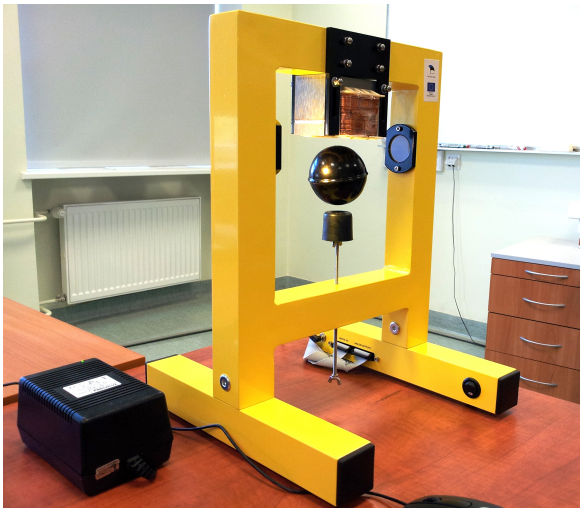


Figure 7.11: Real-life laboratory model of the MLS

Identification of $i(u)$ is relatively simple and straightforward is done with the sphere removed from the MLS, since only the coil current is measured. We obtain the following polynomial:

$$i(u) = -0.3u^2 + 2.6u - 0.047. \tag{7.23}$$

In addition, the deadzone in control is found to be $u_{dz} = [0, 0.0182]$.

Determination of $c(x_1)$ and parameters f_{ip1} and f_{ip2}, on the other hand, is more involved. Because MLS is open-loop unstable, only closed-loop identification is applicable. Our approach is to use the existing PID control loop with

$$K_P = -39, \quad K_I = -10, \quad K_D = -2.05 \tag{7.24}$$

provided by INTECO. It should be noted that a constant input $u_c = 0.38$ is added to the control law $u_{PID}(t)$ in (7.24), that is, the full control law $u(t)$ is such that

$$u(t) = u_{PID}(t) + u_c. \tag{7.25}$$

In order to determine the values of the parameters, we employ time domain simulations and minimize the model output error by means of the least-squares method. The results are as follows:

$$c(x_1) = 3.9996x_1^4 + 3.9248x_1^3 - 0.34183x_1^2 + 0.007058x_1 + 2.9682 \cdot 10^{-5} \tag{7.26}$$

and

$$f_{ip1} = 1.1165 \cdot 10^{-3}\text{m/s}, \quad f_{ip2} = 26.841 \cdot 10^{-3}\text{m}. \tag{7.27}$$

The results of the identification are presented in Figure 7.12. It can be seen that a close fit to the response of the original response of the system is achieved.

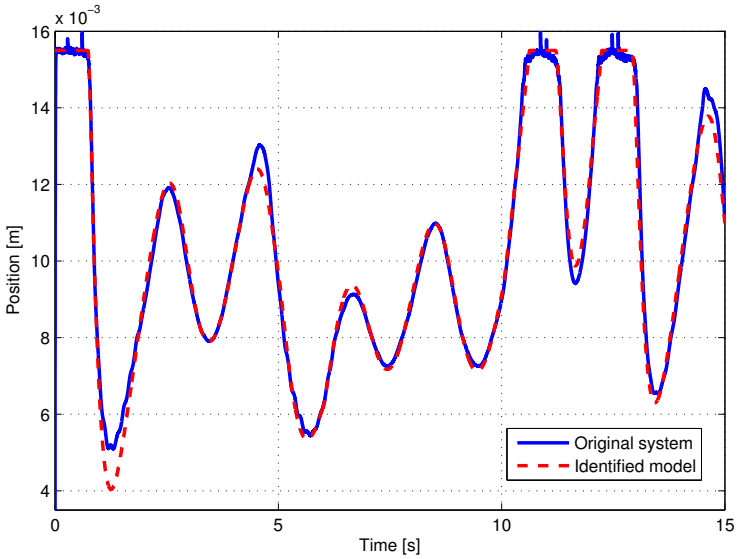


Figure 7.12: Results of nonlinear model parameters identification

7.2.2 FOPID Controller Design for the MLS

We first obtain a linear approximation. We choose a working point $u_0 = 0.3726, x_{10} = 9.84 \cdot 10^{-3}$ and obtain

$$G_M(s) = -\frac{1788}{s^3 + 34.69s^2 - 1737s - 60240}. \tag{7.28}$$

Then, we randomly generate FOPID controllers using the ranges $K_p \in [-100, 0]$, $K_i \in [-50, 0]$, $K_d \in [-25, 0]$, $\lambda \in [0.8, 1.2]$, $\mu \in [0.5, 1.0]$. On the average, about 20 out of 100 tested controllers are found to produce a stable closed-loop system. After inspection, three of them are selected for the optimization phase:

$$C_1(s) = -42.8642 - \frac{18.5653}{s^{1.06}} - 3.0559s^{0.94}, \tag{7.29}$$

$$C_2(s) = -54.3649 - \frac{47.6078}{s^{0.82}} - 6.5436s^{0.98}, \tag{7.30}$$

$$C_3(s) = -45.3118 - \frac{4.24932}{s^{0.86}} - 3.51115s^{0.98}. \tag{7.31}$$

For each controller in this set, we find stability boundaries in different parameter planes, that is, in (K_p, K_i), (K_p, K_d), and (K_i, K_d), so that we can

obtain a wider set of results. Using the stability boundary determination method with a step of $\Delta p = 1$ and choosing a maximum of $N = 20$ steps we locate the following bounds:

$$K_p^{C_1} \in [-62, -34], \quad K_i^{C_1} \in [-38, -1], \tag{7.32}$$

$$K_p^{C_2} \in [-74, -35], \quad K_d^{C_2} \in [-26, -3], \tag{7.33}$$

$$K_i^{C_3} \in [-24, -1], \quad K_d^{C_3} \in [-23, -2]. \tag{7.34}$$

We then proceed directly to the optimization procedure. The FOMCON toolbox FOPID optimization tool is used. We set the bounds of controller gains as in (7.32)–(7.34) for each controller and optimize only the corresponding gains. The number of iterations is, in general, limited to $N_{iter} = 5$. After optimization, the following controllers are obtained:

$$C_1^*(s) = -45.839 - \frac{18.504}{s^{1.06}} - 3.0559s^{0.94}, \tag{7.35}$$

$$C_2^*(s) = -54.444 - \frac{47.6078}{s^{0.82}} - 3.7773s^{0.98}, \tag{7.36}$$

$$C_3^*(s) = -45.3118 - \frac{4.916}{s^{0.86}} - 2.9074s^{0.98}. \tag{7.37}$$

7.2.3 Experimental Results

In the following, we provide the results of performance evaluation of both the randomly generated FOPID controllers, and the suboptimal ones. The controllers are evaluated in a two-cascade closed control loop as detailed in Section 4.4. The parameters of the retuning controllers are computed by means of (4.38) and (4.39). The performance of FOPID controllers is compared to the performance of the original PID control loop, where the parameters of the PID controller are equal to those in (7.24). The reference set point is $x_r = 0.010$m, and a disturbance impulse is considered, appearing for 200ms on the 10th second of the simulation. With the conventional PID controller the following results are achieved:

$$\tau_s = 3.34\,\text{s}, \quad \theta = 66.0\%, \quad \theta_d = 60.6\%.$$

In Table 7.3 the performance evaluation of the FOPID controllers working in the retuning control loop is presented. It can be seen, that the best performance is achieved when controller $C_2^*(s)$ is used. The result of real-time simulation of this controller versus the original PID control loop is provided in Figure 7.13. It can be seen that a significant improvement in control system response is obtained. The controller $C_3(s)$ outperforms the original PID only in terms of overshoot, while $C_3^*(s)$ offers similar settling time with a much smaller overshoot.

In addition, we consider a reference tracking experiment to illustrate the ability of the controllers to provide appropriate regulation across a wider operating range. The comparison of the performance of the $C_2^*(s)$ controller and the original control loop is presented in Figure 7.14. Once again, improvements in the control loop performance can be observed.

Table 7.3: Comparison of FOPID controller performance

FOPID	τ_s[s]	θ[%]	θ_d[%]	FOPID*	τ_s[s]	θ[%]	θ_d[%]
$C_1(s)$	1.85	24.0	60.3	$C_1^*(s)$	1.68	14.8	56.4
$C_2(s)$	**1.39**	**19.4**	**37.5**	$C_2^*(s)$	**0.86**	**11.6**	**34.6**
$C_3(s)$	4.68	14.6	55.7	$C_3^*(s)$	3.84	15.0	58.3

7.3 Control of Ion-Polymer Metal Composite Actuator

Ionic polymer-metal composite (IPMC) actuators are electroactive polymer (EAP) materials that change their shape or size in response to electrical excitation [10]. The same material may be used as a sensor, since external deformations of the laminate result in the generation of current [102]. Potential applications of IPMC actuators include soft robotics [78], biomedicine [104], and space [103].

In general, long-term low-frequency characteristics of IPMC actuators exhibit nonlinear effects, such as back-relaxation [10]. Several approaches were considered for the task of controlling the IPMC actuator, e.g., [139,140]. In this work we are concerned only with the dominating dynamics. The problem of compensating back-relaxation in non-ideal conditions falls outside of the scope of this thesis.

The displacement of the conventional water-based IPMC actuator under a voltage signal apparently exhibits FO dynamics [16, 17, 18]. Some initial results on FO control of IPMC actuators were reported in [54]. Therefore, it appears that FO identification and control of our novel IPMC actuator with nanoporous carbon-based electrodes is natural and favorable, since FO calculus provides a more accurate description for materials, which are known to possess memory-like phenomena. In this work, we verify the benefits of FO calculus-based modeling and control approach on carbon-based IPMC actuator by achieving displacement control without feedback.

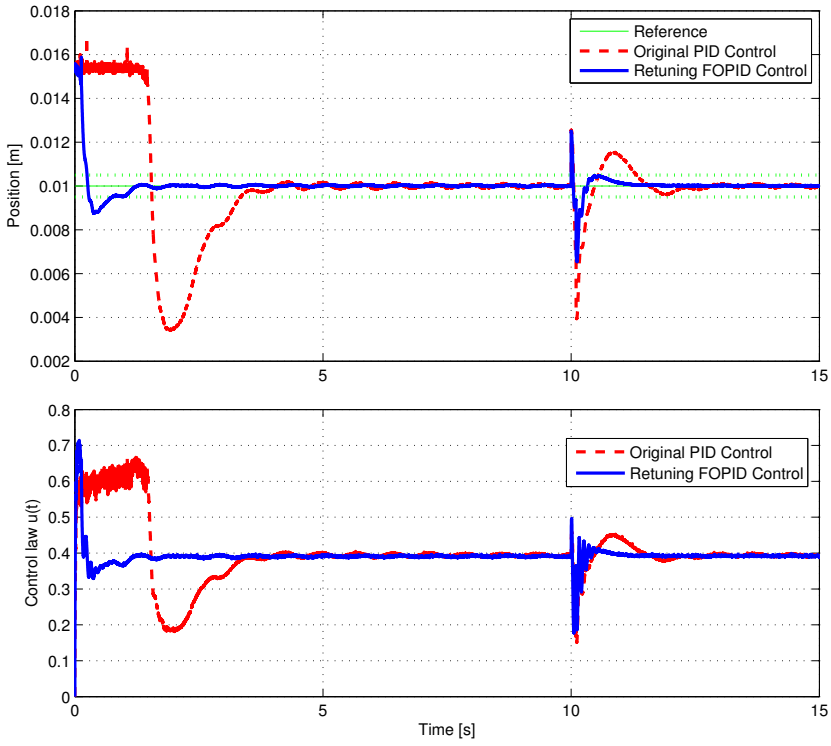


Figure 7.13: MLS control system step experiment: Original PID control vs. Retuning FOPID controller

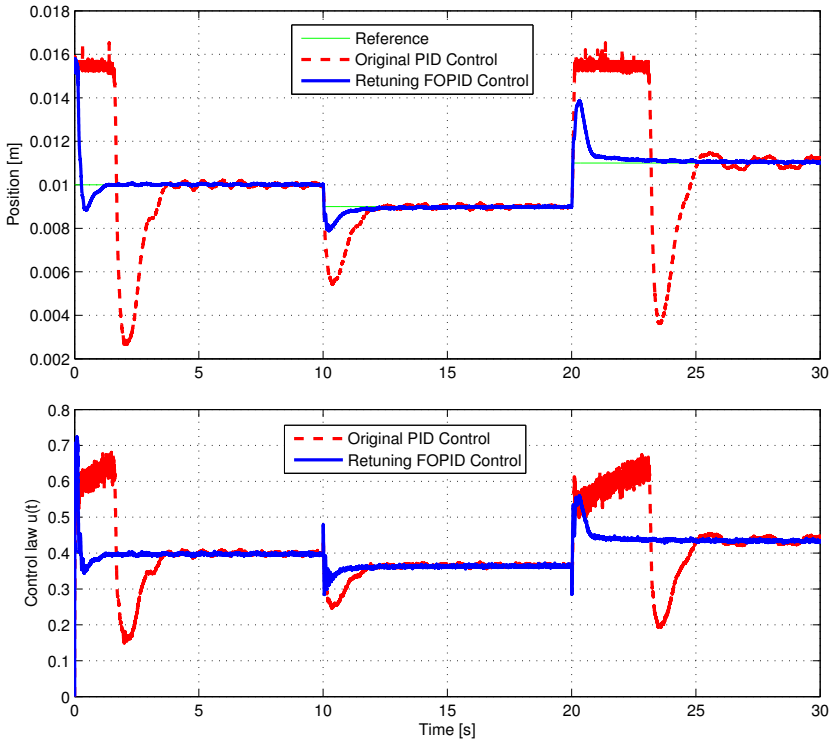


Figure 7.14: MLS control system reference tracking: Original PID control vs. Retuning FOPID controller

IPMC Actuator Description

The IPMC actuator demonstrated in this study was manufactured in Intelligent Materials and Systems laboratory in University of Tartu. It comprises 5 symmetrically aligned layers—a Nafion ion-polymer membrane, which is covered from both sides with porous carbon electrodes further covered with gold (see Figure 7.15). The gold foil increases the originally moderate electrode conductivity and allows electric current to reach outermost parts along the actuator. As a result, the actuator bends almost uniformly all over the sample. The electrodes are spray-painted using direct assembly process first introduced in [1]. Details about the manufacturing process are given in [94]. Initially, the composite is manufactured in 50 x 50 mm^2 patches, from which a 5 x 20 mm^2 sample with total thickness of ∼300 µm was cut for experiments.

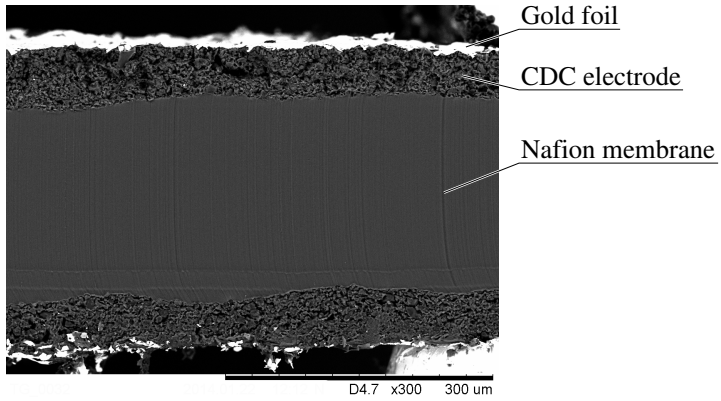


Figure 7.15: Scanning electron microscope (SEM) image of cross-section of carbon-based IPMC actuator

Experimental Setup

The experimental setup, located in the Intelligent Materials and Systems laboratory in University of Tartu, allows one to remotely connect to the host computer via a secure connection and to perform the necessary experiments for identification and control of the system under investigation. The experimental schematic is provided in Figure 7.16. The IPMC actuator is operated from MATLAB/Simulink environment through National Instrument data acquisition devices (PCI-6034E, PCI-6703). The OPA548 operational amplifier in combination with the 20 Ω shunt resistor takes care of supplying enough electrical current to the actuator and is also used for current monitoring purposes. The performance of the actuator is determined through a BANNER LG10A65PU laser sensor, which is set to register the transverse displacement at 15 mm away from the input clamps.

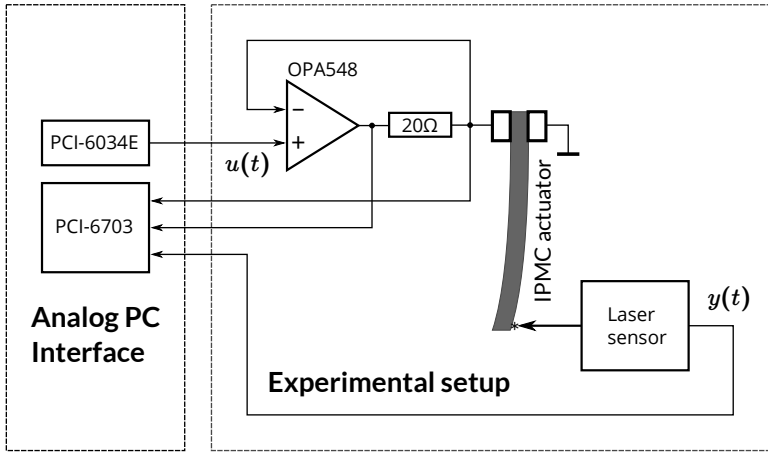


Figure 7.16: Experimental schematic

It has been shown that IPMC actuators may be highly sensitive to humidity fluctuations [79]. In this work we are not concerned with the impact of humidity on the actuator. Therefore, it was placed into a clean, nitrogen-filled environment, from which traces of H_2O and O_2 were almost completely removed (0.01 ppm and 0.2 ppm, respectively). Due to the capacitive nature of IPMC actuators, one could easily damage the actuator by overheating it by supplying a high voltage signal or by rapid signal switching. Thus, in all experiments the voltage signal is limited to the range $u_V = (-1.5, 1.5)$ V, which, based on prior experiments, is assumed to be safe for relatively long-term experiments. Additionally, one can place a low-pass filter at the output of the controller for limiting high-frequency switching and noise.

7.3.1 Identification of the Actuator Model

During our experiments the IPMC actuator achieved steady state under a constant voltage signal with a relatively small back-relaxation effect, which allows to use time domain identification to capture the essential dynamics of the actuator, and to design a suitable compensator to control it. Once the identification data is collected (Figure 7.17) from the IPMC plant it becomes evident that the dynamical properties of the system resemble that of a classic, first-order process model. In order to verify this, identification process is carried out, where the structure of the linear model is

$$G_{IO}(s) = \frac{K}{Ts + 1}. \tag{7.38}$$

However, one may also note that the output of the system is converging to the final state slowly. This points to the possibility of fractional-order dynamics. Thus, a model of the form (2.14) is also identified. Note that the

initial state—the displacement distance of the material measured by the laser sensor—is considered to be an offset, and is removed from the identification data and is subtracted from the output during real-time control experiments.

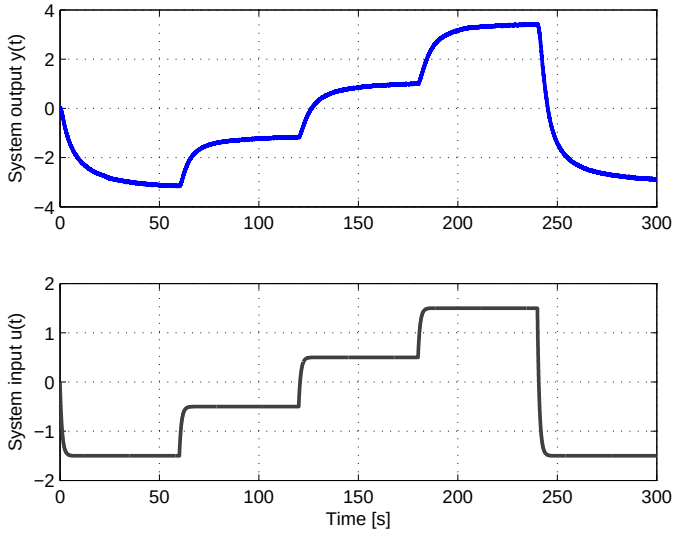


Figure 7.17: Identification dataset

The following models are obtained using the time domain identification tool of FOMCON toolbox:

$$G_1(s) = \frac{2.0688}{7.5959s + 1} \tag{7.39}$$

and

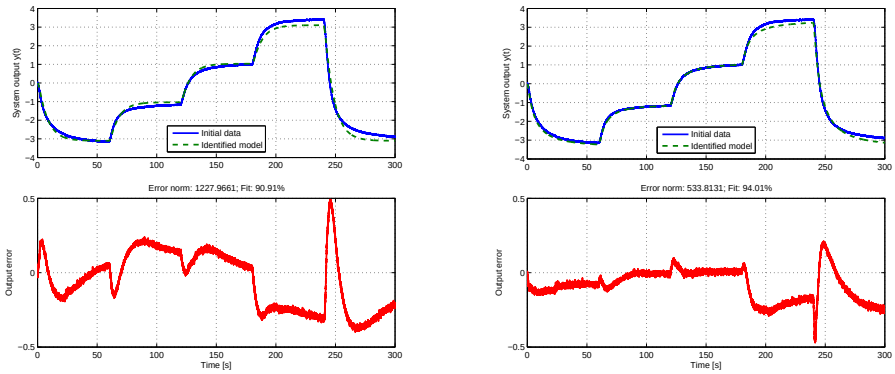
$$G_2(s) = \frac{2.2993}{4.8317s^{0.7797} + 1}. \tag{7.40}$$

The comparison of the quality of the models in replicating the behavior of the system in the time domain is given in Figure 7.18. For measuring the quality of the model we use the residual error norm:

$$\xi = \sum_{i=1}^{n} \varepsilon_i^2 = \|\varepsilon\|_2^2 , \tag{7.41}$$

It can be clearly seen that the FO model provides a better description of the process in the whole operating range with an error norm of $\xi = 533.8086$ and a fit of 94.01% to experimental data, while the IO model has an error norm of $\xi = 1227.9659$ and a fit of 90.91%. One can also note that the system appears to be adequately described by linear models.

We consider two types of controllers for PC based control. First, we consider using a FOPID controller. Second, the use of fractional inversion model based control is investigated.



(a) Integer-order model of the IPMC actuator

(b) Fractional-order model of the IPMC actuator

Figure 7.18: IPMC actuator: Identification results

7.3.2 FOPID Control

Since feedback from the object will likely not be available in a real-life application, a reference model is used for control. The control configuration is given in Figure 7.19.

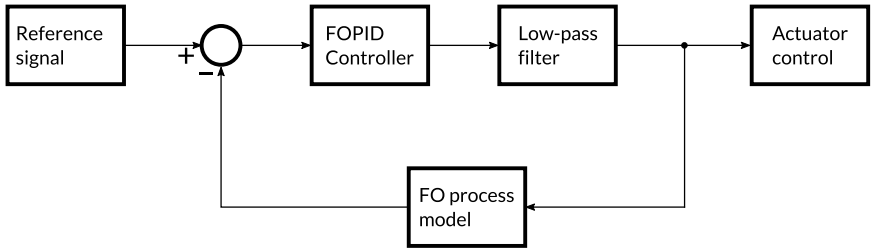


Figure 7.19: FOPID control via reference model

In this work, the following control system design specifications are considered. In the time domain, the IAE performance index in (4.3) is chosen, while in the frequency domain we consider the following specifications

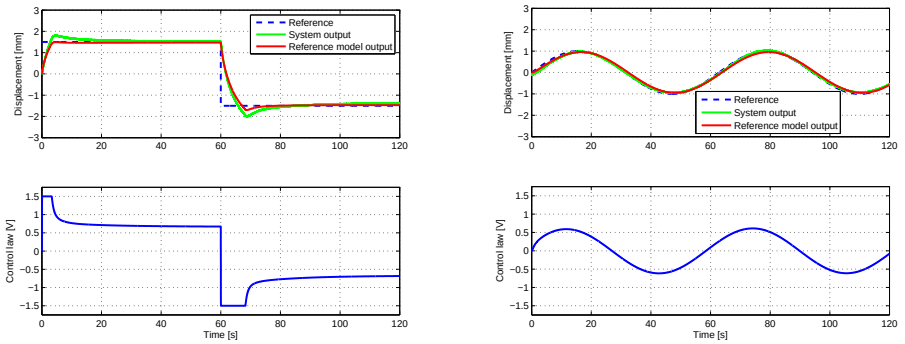
- Phase margin φ_m and the corresponding crossover frequency ω_c;
- Robustness to gain variations requirement in (4.8).

The model in (7.40) is used for both control design and as the reference model. By varying the phase margin and critical frequency it is possible to

achieve a quality balance between robustness of the control system to external disturbances and the amount of overshoot/undershoot during setpoint transitions. We choose $\varphi_m = 100^\circ$ and $\omega_c = 1.25$. Initial analysis revealed that under current conditions using a PI^λ controller is sufficient, so three design parameters, i.e., $\{K_p, K_i, \lambda\}$ are tuned using the FOPID optimization tool from FOMCON toolbox. The following controller was designed subject to the specifications given above

$$C(s) = 1.7887 + 1.3617 s^{-0.51513}. \tag{7.42}$$

The results of real-time control experiments with the IPMC actuator are given in Figures 7.20a and 7.20b for piecewise constant reference and for sinusoidal reference signals, respectively.



(a) Piecewise constant reference

(b) Sine wave tracking

Figure 7.20: Personal Computer-based FOPI control of the IPMC actuator

A relatively large overshoot is observed in case of a piecewise constant tracking. The performance indices for piecewise constant tracking and sine wave tracking are $IAE_{pc} = 18.3471$ and $IAE_{sine} = 5.7663$, respectively.

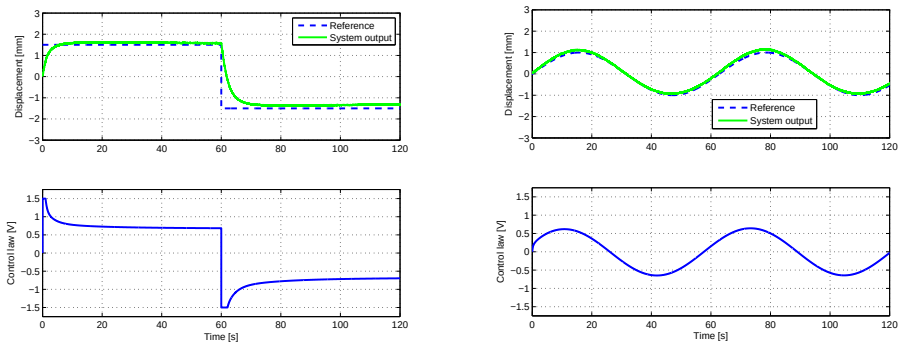
7.3.3 FOINVM based Control

The controller of the form (2.31) is due to (7.40) and is given by the FO transfer function

$$C(s) = \frac{1 + 4.8317 s^{0.7797}}{2.2993(1 + 0.1s)}. \tag{7.43}$$

The results of real-time control experiments are given in Figures 7.21a and 7.21b for piecewise constant reference and for sinusoidal reference signals, respectively.

The performance indices for piecewise constant tracking and sine wave tracking are $IAE_{pc} = 23.3436$ and $IAE_{sine} = 8.3605$, respectively.



(a) Piecewise constant reference

(b) Sine wave tracking

Figure 7.21: Personal Computer-based fractional inversion model based control of the IPMC actuator

Since the FOPI controller exhibits better performance in terms of overall control quality based on the IAE measure, it is used in the following experiments for the hardware implementation of the controller.

7.3.4 Hardware Implementation of the Controller

In the following, hardware implementation of the designed FOPI controller is discussed. The experimental platform from Section 5.4 is used with the advanced hardware prototype of the FOPID controller from Section 5.5.2. The older prototype based on the Atmel AVR ATmega8A/324 controller was initially considered. However, it does not have enough computational accuracy/performance to be efficiently used as a FOPID controller with an integrated reference model.

Two types of experiments are carried out:

- Simulation only—the performance FOPID controller prototype with internal reference model is verified;
- Experiment with the real object—the reference signal and the corresponding control law due to the open-loop controller are sent to the laboratory in Tartu via a secure connection.

The schematic diagram for the latter experiment is given in Figure 7.22. The admissible control range of IPMC actuator displacement is chosen to be $y_a = [-5, 5]$ mm and all the analog signals are scaled accordingly.

The software implementation of the FOPID controller is as follows. A digital filter approximation of the reference model in (7.40) is obtained in

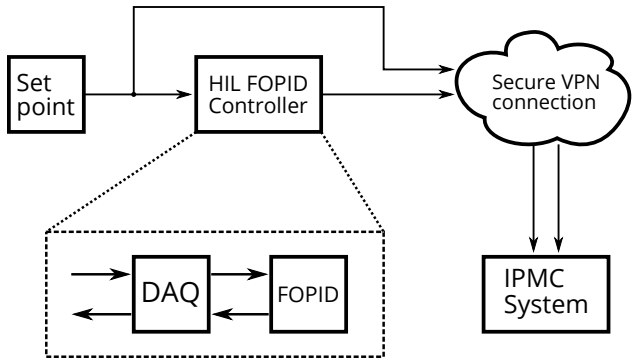


Figure 7.22: Open-loop control of IPMC actuator over the network using the HIL prototype

IIR SOS form with the following coefficients

$$\begin{aligned} b &= \{-0.0000000000, +0.0000000000\}, \\ &\quad \{-0.0292603776, +0.0000001205\}, \\ &\quad \{-1.1166551079, +0.2746400665\}, \\ &\quad \{-1.8986853766, +0.9004848473\}, \\ &\quad \{-1.9915211928, +0.9915336126\}, \\ &\quad \{-1.9993104973, +0.9993105794\}, \end{aligned} \tag{7.44}$$

$$\begin{aligned} a &= \{-0.0000000785, +0.0000000000\}, \\ &\quad \{-0.2734226971, +0.0024889370\}, \\ &\quad \{-1.5774636716, +0.6108120894\}, \\ &\quad \{-1.9555485172, +0.9559265941\}, \\ &\quad \{-1.9936348081, +0.9936426878\}, \\ &\quad \{-1.9993309003, +0.9993309786\}, \end{aligned} \tag{7.45}$$

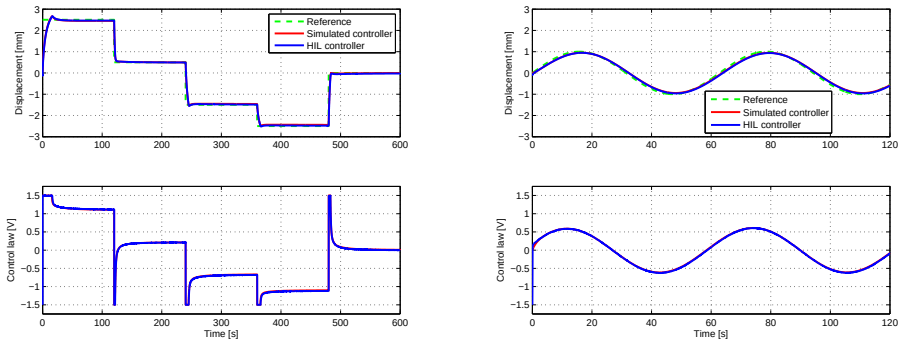
and

$$b_0 = 0.0454022102. \tag{7.46}$$

The reference model is hard-coded in terms of static IIR SOS coefficient arrays given in Equations (7.44), (7.44), and (7.46), whereas the IIR filters in SOS form for the controller itself are generated in real time using the method from Section 5.3.

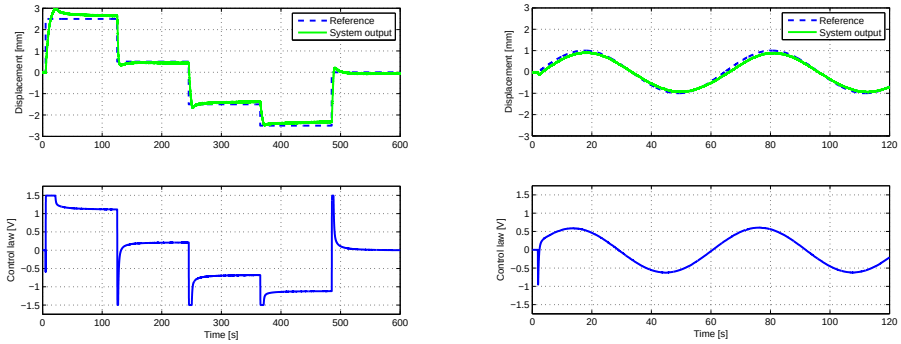
The behavior of the IPMC actuator is investigated in an inert environment as before. The results of the experiments are given in Figures 7.23a and 7.23b for local simulation, and in Figures 7.24a and 7.24b for the network control experiment with the real IPMC actuator. Control error arising in

the case of piecewise constant signal is provided in Figure 7.25. As it can be seen, the controller prototype is successfully experimentally validated, which means that the computed reference model is long-term stable, and open loop control in case of the real object is sufficiently accurate. The performance indices for the inert environment are $IAE_{pc}^{gb} = 85.3902$ and $IAE_{sine}^{gb} = 9.6365$.



(a) Piecewise constant set point (b) Sine wave tracking

Figure 7.23: FOPI control: HIL controller with simulated system



(a) Piecewise constant set point (b) Sine wave tracking

Figure 7.24: FOPI control: HIL controller with real system

7.4 Conclusions

The following general conclusions can be made based on the results given in this chapter.

- Fractional dynamics were observed in process control applications. Thus,

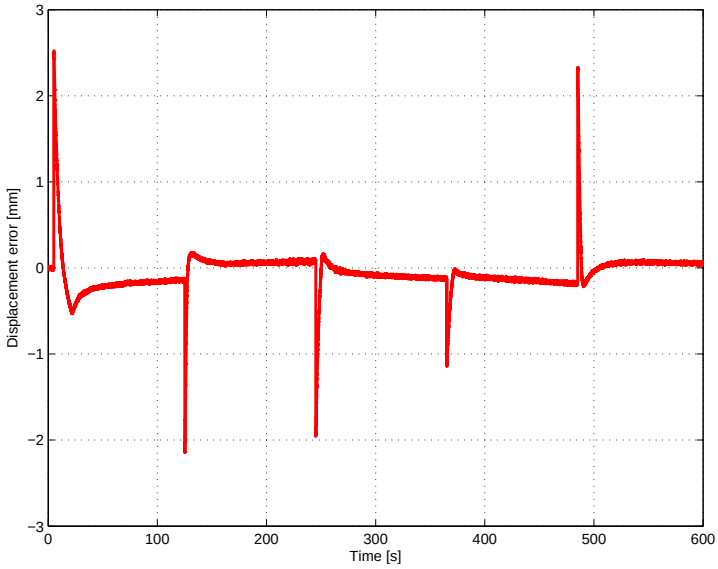


Figure 7.25: FOPI control error: HIL controller with real system, piecewise constant set point

one may conclude that using FOPID controllers instead of PID controllers will lead to improvement of control loop performance in studied cases, since only such controllers are capable of compensating for FO dynamics.

- The flexibility of FOPID controllers was illustrated by the ability to fulfill more design specifications and by the larger stability zone for stabilizing unstable plants.
- The two-point gain and order scheduling and control loop retuning methods were successfully verified, showing their applicability to real-life control problems;
- The advanced hardware prototype of the FOPID controller was validated in real-time experiments. It showed expected performance and maintained long-term stability, from which it can be concluded that the implementation was successful.

Specific comments pertaining to each control object discussed in the chapter are given next.

Multi Tank System

In case of the coupled tanks system, we have obtained results comparable to those achieved by nonlinear control in [41, 141] with the important difference

that FOPID controllers ensure control system robustness in the presence of disturbances. However, the difference in the model and the real plant is quite evident and should be eliminated. A more thorough parameter identification procedure may be needed. Once a more accurate model is obtained, the results of controller design via simulation based optimization should also improve.

In case of the Multi Tank system from INTECO, an efficient control method involving a composite control law comprised of FOPID controllers was successfully applied to the control problem. The proposed method is quite simple, requires only static description of the FOPID controllers and therefore may be employed in, e.g., automatic tuning for efficient control of nonlinear systems with across a large operating range. However, we perform only heuristic linear stability analysis of the resulting composite control system, but it would be more beneficial to consider stability analysis of the nonlinear system. In addition, further work may be carried out to design a more efficient controller for the middle tank, such that would minimize the switching of automatic valves. Finally, due to considerable measurement noise an extended Kalman filter was designed since linear filters did not provide acceptable performance.

The prospective use of FOPID controllers lies in the industrial domain, where the majority of control loops are based on PI and PID controllers [6,83], including those dedicated to fluid level control. It can be concluded that application of FOPID controllers to the problem of level control is justified, since such controllers offer more tuning flexibility and allow taking into account more robustness criteria. In case of disturbances, measurement noise, effects of fractional dynamics as well as other unmodeled dynamics of real industrial systems, even small gains in performance arising from using FOPID controllers in place of conventional PID controllers will lead to an overall improvement, since typical industrial applications comprise many control loops [149].

Magnetic Levitation System

In this application, an unstable plant, namely the MLS system was considered. A nonlinear model of this plant was identified from a closed-loop experiment. Linear analysis methods were employed to determine stabilizing FOPID controllers and stability boundaries in two-dimensional parameter planes thereof. The controllers were then evaluated, and those with the best performance were optimized. In all cases, the optimization procedure enhanced the performance of the control loop. Virtually all retuning controllers offer superior performance compared to the original control loop thereby establishing the validity of the proposed approach, which can also be applied to other nonlinear systems.

These results illustrate the benefits of using a FO controller to stabilize an unstable plant. Unfortunately, a serious limitation was found, that is the difficulty to impose robustness specifications in case of FOPID control of MLS. While there are more opportunities to stabilize the plant, it is difficult to guarantee noise immunity and disturbance rejection of the designed control loop. Towards that end, the adoption of results from [82] and [56] may lead to the solution of this problem.

IPMC Actuator

In what follows, conclusions are drawn based upon the conducted series of experiments. The first batch of experiments was carried out using MATLAB/Simulink for identification and control, while the second was done using a HIL approach with a FOPID controller prototype.

First, based upon experimental evidence gathered thus far, it appears that the IPMC with carbon-based electrodes may be adequately modeled by FO differential equations, hence it is natural to use FO controllers for control thereof. Earlier results also indicate this [16, 18].

In this work, we were concerned with the principal dynamics of the system. In an inert environment the relaxation effect was minimal and was not taken into consideration. Relaxation dynamics appear to be of fractional nature, hence a more complicated model may be used to account for this effect [18]. This is also one of the reasons to apply fractional control methods and not those based on nonlinear control system analysis.

Two types of controllers were considered: FOPID controller and FOINVM controller. The following comments pertain to the achieved control performance in the first series of experiments.

- FOPI controller exhibits superior performance than the FOINVM controller;
- Both controllers rely on the quality of the reference model, however:
  - In case of the FOPID controller, some robustness criteria are considered, which means that it should perform better than FOINVM under disturbances and noise, if feedback of some form is present. If IPMC actuators are to be used in a macro-object, feedback may be available, albeit in a different form, e.g., as the displacement of the macro-object, the latter set to perform a particular task;
  - To improve overall control quality it may be necessary to consider gain scheduling, that is, design a number of controllers for different working points of the system.

The following comments are related to the second series of experiments, where a hardware implementation of a FOPI controller was used due a prototype assembled in Alpha Control Laboratory.

- Relatively accurate control of the process was observed during the network-based experiments. Because no feedback is used in control, the communication delay may be neglected.
- The internal reference model is currently hard-coded into the prototype. This means the only direct means to update it is through changing its static gain. For changing the dynamics, a more intricate procedure is needed.
- The FOPID prototype used in these experiments is relatively large, since it is meant for industrial-grade experiments. In terms of integration, it is possible to make the microcontroller-based FOPI prototype PC board much smaller (about the size of a 2€ coin). Also, other implementation methods exist and may be used, e.g., analog modeling of FOPID controllers considered in which allow for a much smaller scale.

By conducting a series of experiments over a course of several weeks, we have found that the experiments in the inert environment are repeatable and model parameter variations are reasonably small. All considered controllers use reference model-generated feedback, which means the quality of control strongly depends on the quality of the reference model. In a real-life environment the model must be updated in real time to reflect the changing conditions, such as humidity, temperature, etc. This requires further investigation.

Conclusions

> Every great and deep difficulty bears in itself its own solution. It forces us to change our thinking in order to find it.
>
> *Niels Bohr*

> The scientific man does not aim at an immediate result. He does not expect that his advanced ideas will be readily taken up. His work is like that of the planter—for the future. His duty is to lay the foundation for those who are to come, and point the way.
>
> *Nikola Tesla*

We now formulate concluding remarks pertaining to the results presented in this thesis. Accordance thereof to the problems stated at the beginning of the dissertation is established.

First, the problem of identification of systems by fractional models was considered. Primary focus was on time domain identification methods. Several optimization algorithms were used to identify the parameters of fractional models of dynamic systems based on nonlinear least-squares parameter estimation. In addition, some methods for the analysis of the resulting models were discussed. Several solutions to the general problem of efficient identification of fractional models—estimation of commensurate order models—were proposed, mostly based on choosing different subsets of parameters to estimate, including parametric identification, which is also useful in solving closed-loop identification problems. The algorithms were verified by considering several exemplary fractional systems. The optimization methods were shown to be effective in solutions of specific problems. Another important issue encountered is the selection of the identified model structure, that is, the *initialization* of the identification problem and must be further studied. It may be necessary to introduce changes to the optimization algorithms to

tackle this issue.

The choice of the identification algorithm depends on the available data. A carefully chosen excitation signal in case of time domain identification is key to obtaining an accurate model of the system under study. However, conducting such experiments is rarely possible in case of industrial systems, since they are associated with considerable costs. This is why frequency domain methods are of great practical value—especially the available relay feedback based methods—as the output of the industrial system will only slightly deviate from the set point during the identification experiment. Another possibility is to employ closed-loop identification. Since industrial processes are generally complex, they may exhibit phenomena such as long-range dependence and self-similarity. The proposed methods are designed for fractional-order systems and can therefore be applied to modeling such phenomena in the industrial context to obtain more accurate models.

Next, FOPID controller design methods were discussed. A general Nelder-Mead simplex method based optimization algorithm was proposed subject to control system design specifications in both time- and frequency domains. Since the conventional version of this optimization algorithm can only solve unbounded and unconstrained problems, a modification of the method is considered, where frequency domain specifications are formulated as penalty functions for the cost function. Different subsets of FOPID controller parameters may be considered for optimization. The effectiveness of this approach was shown in a motivating example, where controller gains were chosen using a classical PID controller tuning rule [152], while the proper selection of orders of the controller integral and differential components achieved by means of optimization thereof resulted in a significant improvement of control loop performance. Then, the gain and order scheduling approach was discussed. A particular case, when it is sufficient to choose two operating points to effectively control the system under study in the full control range, was investigated. A method for analyzing the stability of the resulting FOPID control system in the linear sense was proposed. This method shared the main idea with the one proposed next in the same chapter, where stability boundaries were heuristically located for a closed-loop control system comprising a FOPID controller and an unstable plant. Since the method is heuristic, efficiency of involved computations cannot be guaranteed. Finally, a conventional PID controller control loop retuning method was presented, such that allows incorporating fractional-order dynamics into existing control loops without internal modifications.

An important conclusion is that since FOPID controllers are technically extensions of conventional PID controllers, existing tuning algorithms for the latter may be adopted for FO control, if necessary. Thus, an engineer may select the best performing controller for a particular industrial control task. If system dynamics are modeled reasonably well with classical FOPDT

models, it is expected that using PID controllers is sufficient.

Final parts of the first two chapters also include two particular methods for identification and control of dynamic systems described by a FFOPDT model. These methods are applicable in process control. First, a frequency domain identification method was proposed using a relay feedback test to determine several frequency response points and to estimate a FFOPDT model based on that information. Conditions under which this is possible were derived. The estimation algorithm was successfully verified on a set of FFOPDT models. However, only a nominal delay value of the model was considered. Further, all frequency domain characteristics of the FFOPDT model were derived, using a Newton-Raphson method specifically tailored for the task. Then, all frequency domain characteristics of a control system comprising a FOPID controller and a FFOPDT model were derived. A FOPID controller gain optimization method based on design specifications in the frequency domain was proposed based on the Newton's method in several dimensions. The effectiveness of this method was demonstrated on an exemplary system.

From the evidence presented in this thesis—specifically in Chapter 7—we may conclude that the proposed method may indeed be of use in an industrial setting, since it takes into account fractional dynamics of a process under study. In addition, the obtained results may be used in automatic control of heating, ventilating, and air conditioning systems in Smart Houses.

In the chapter devoted to implementation of fractional systems and controllers several problems were discussed. First, an updated Carlson method for approximation of fractional capacitors was proposed. It appears to be especially useful in approximating FOLLC controllers. However, it does require a sufficient amount of computational accuracy and resources. Next, a unified approach to design of fractance networks in terms of analog filters comprising passive (resistors, capacitors, and inductors) and active (operational amplifiers) was proposed. The approach relies on the implementation of a programming paradigm—the fractance network has an abstract object class, which incorporates all the necessary properties and methods to work with the class and to obtain the network approximations. Finally, a method for efficient digital implementation of FOPID and FOLLC controllers was described. For verification of the results in this chapter, a real-time prototyping platform was used, based on available hardware and software. The hardware FOPID controller prototypes were built, programmed, and successfully tested by means of this platform.

The main issue with the implementation of fractional models—the difference in magnitude of the parameters of the resulting approximations—was solved for both analog and digital realizations. However, the solution requires either a complicated analog circuit manufacturing process, or relatively high-end hardware for the digital implementation. Therefore, this issue must be

investigated further.

Most of the results described in Chapters 3 through 5 are incorporated into FOMCON toolbox for MATLAB/Simulink. In the corresponding chapter, the features of the toolbox were presented and a variety of illustrative examples, some of which extend the ideas developed in previous chapters, were given.

Finally, a chapter devoted to practical application of fractional control deals with three models of industrial control objects (two types of fluid tank systems and a magnetic levitation system), as well as with the problem of control of a novel IPMC actuator. The identification, control, and control system implementation methods discussed in previous chapters were successfully applied, showing the benefits arising from the extended modeling and control flexibility offered by fractional models. Moreover, the advanced FOPID controller hardware prototype was successfully verified during the experiments with the IPMC actuator.

These results demonstrate the applicability of FO control to real-life problems. The decision to use FOPID controllers is dictated by universal use of PID controllers in the industry. Comparable nonlinear control methods are available, but offer less general means for efficient control [149], whereas FOPID controllers are more flexible, provide more opportunities for stabilizing unstable plants, and can work in conjunction or in parallel with conventional PID control loops, which generally leads to improvement of performance of the overall system.

Future Research

During the thesis work, some issues were identified. These issues form the basis for future research. A more specific summary is provided next.

One of the main obstacles for successful application of the extended stabilization possibilities of fractional controllers is the absence of general results related to stability analysis. This leads to the development of heuristic methods, some of which were presented in this work, which are not computationally efficient. It is thus of significant interest to investigate and develop efficient general methods for analyzing the stability of fractional systems. Towards that end, it may be possible to extend the results in, e.g., [82] or [56]. Such results may be further applied in model based control design.

In this work, mostly SISO control problems were considered. However, real-life industrial control problems usually comprise multiple loops. Therefore, a more sophisticated and generalized retuning approach for existing PI/PID control loops should be developed. Also, the automatic tuning procedure for multiple control loops should be appropriately treated [149].

In addition, a general scheme for embedded implementation of fractional

systems should be investigated. In this work we have covered FOPID and FOLLC controller generation. However, the reference model was not generated on the prototype, but rather the SOS coefficients of IIR filters were hard-coded into it. In addition, variable-order operators may be employed in the implementation of FOPID controllers. This will open up even further opportunities for the realization of efficient and accurate automatic tuning algorithms.

The algorithms for system identification and controller design developed for process control applications must be verified using the advanced hardware FOPID controller prototype across a sufficient number of industrial plant models and real-life objects to ensure industry-grade performance and reliability.

It is the firm belief of the author that using generalized fractional models in system theory is a natural step towards more accurate modeling and developing efficient control systems, which the results of this thesis seem to indicate. Therefore, using fractional models and controllers is expected to become part of standard practice in the coming years.

Bibliography

[1] B. Akle, M. Bennett, D. Leo, K. Wiles, and J. McGrath, "Direct assembly process: a novel fabrication technique for large strain ionic polymer transducers," *Journal of Materials Science*, vol. 42, no. 16, pp. 7031–7041, 2007.

[2] N. Al-Muthairi and M. Zribi, "Sliding mode control of a magnetic levitation system," *Mathematical Problems in Engineering*, vol. 2004, no. 2, pp. 93–107, 2004.

[3] K. H. Ang, G. Chong, and Y. Li, "PID control system analysis, design, and technology," *Control Systems Technology, IEEE Transactions on*, vol. 13, no. 4, pp. 559–576, July 2005.

[4] K. Åström and T. Hägglund, *PID Controllers: Theory, Design, and Tuning*, 2nd ed. The Instrumentation, Systems, and Automation Society (ISA), 1995.

[5] K. Åström and T. Hägglund, "The future of PID control," *Control Engineering Practice*, vol. 9, no. 11, pp. 1163 – 1175, 2001.

[6] K. Åström and T. Hägglund, *Advanced PID control*. The Instrumentation, Systems, and Automation Society (ISA), 2006.

[7] K. Atkinson, *An Introduction to Numerical Analysis*, 2nd ed. Wiley, 1989.

[8] Atmel Corporation. (2015) Official website of Atmel Corporation. [Last access time: 12.05.2015]. [Online]. Available: http://www.atmel.com/

[9] V. Bandal and P. Vernekar, "Design of a discrete-time sliding mode controller for a magnetic levitation system using multirate output feedback," in *American Control Conference (ACC), 2010*, June 2010, pp. 4289–4294.

[10] Y. Bar-Cohen, *Electroactive Polymer (EAP) Actuators as Artificial Muscles: Reality, Potential, and Challenges*. Society of Photo Optical, 2004.

[11] S. Barrett and D. Pack, *Atmel Avr Microcontroller Primer: Programming and Interfacing, Second Edition*, ser. Synthesis lectures on digital circuits and systems 1932-3166 ; lecture #39. Morgan & Claypool, 2012.

[12] J.-L. Battaglia, O. Cois, L. Puigsegur, and A. Oustaloup, "Solving an inverse heat conduction problem using a non-integer identified model," *International Journal of Heat and Mass Transfer*, vol. 44, pp. 2671–2680, 2001.

[13] J. Belikov, S. Nõmm, E. Petlenkov, and K. Vassiljeva, "Application of neural networks based SANARX model for identification and control liquid level tank system," in *The 12th International Conference on Machine Learning and Applications*, Miami, FL, USA, December 2013, pp. 246–251.

[14] K. Binmore, *Mathematical Analysis: A Straightforward Approach.* Cambridge University Press, 1982.

[15] M. A. Branch, T. F. Coleman, and Y. Li, "A subspace, interior, and conjugate gradient method for large-scale bound-constrained minimization problems," *SIAM J. Sci. Comput.*, vol. 21, no. 1, pp. 1–23, Aug. 1999.

[16] R. Caponetto, *Fractional Order Systems: Modeling and Control Applications*, ser. World Scientific Series on Nonlinear Science: Series A. World Scientific, 2010.

[17] R. Caponetto, G. Dongola, L. Fortuna, S. Graziani, and S. Strazzeri, "A fractional model for IPMC actuators," in *Instrumentation and Measurement Technology Conference Proceedings, 2008. IMTC 2008. IEEE*, May 2008, pp. 2103–2107.

[18] R. Caponetto, S. Graziani, F. Sapuppo, and V. Tomasello, "An enhanced fractional order model of ionic polymer-metal composites actuator," *Advances in Mathematical Physics*, vol. 2013, pp. 1–6, 2013.

[19] G. Carlson and C. Halijak, "Approximation of fractional capacitors $(1/s)^{1/n}$ by a regular Newton process," *IEEE Transactions on Circuit Theory*, vol. 11, no. 2, pp. 210–213, 1964.

[20] L.-Y. Chang and H.-C. Chen, "Tuning of fractional PID controllers using adaptive genetic algorithm for active magnetic bearing system," *WTOS*, vol. 8, no. 1, pp. 158–167, Jan. 2009.

[21] A. Charef, H. H. Sun, Y. Y. Tsao, and B. Onaral, "Fractal system as represented by singularity function," *IEEE Transactions on Automatic Control*, vol. 37, no. 9, pp. 1465–1470, 1992.

[22] W. Chen, *Passive, Active, And Digital Filters*. Taylor & Francis, 2006.

[23] Y. Q. Chen, I. Petráš, and D. Xue, "Fractional order control - a tutorial," in *Proc. ACC '09. American Control Conference*, 2009, pp. 1397–1411.

[24] O. Cois, A. Oustaloup, T. Poinot, and J.-L. Battaglia, "Fractional state variable filter for system identification by fractional model," in *Proc. of the 2001 European Control Conference*, 2001, pp. 417–433.

[25] T. F. Coleman and Y. Li, "An interior trust region approach for nonlinear minimization subject to bounds," Department of Computer Science, Cornell University, Ithaca, NY, USA, Tech. Rep., 1993.

[26] F. Declercq and R. De Keyser, "Comparative study of neural predictors in model based predictive control," in *Neural Networks for Identification, Control, Robotics, and Signal/Image Processing, 1996. Proceedings., International Workshop on*, Aug 1996, pp. 20–28.

[27] L. Dorčák, J. Terpák, I. Petráš, and F. Dorčáková, "Electronic realization of the fractional-order systems," *Acta Montanistica Slovaca*, vol. 12, no. 3, pp. 231–237, 2007.

[28] L. Dorčák, J. Valsa, J. Terpák, I. Petráš, and E. Gonzalez, "Comparison of the electronic realization of the fractional-order system and its model," in *Proc. 13th International Carpathian Control Conference (ICCC)*, 2012, pp. 119–124.

[29] L. Dorčák, I. Petráš, E. A. Gonzalez, J. Valsa, J. Terpák, and M. Zecova, "Application of PID retuning method for laboratory feedback control system incorporating fo dynamics," in *Proc. of the 14th International Carpathian Control Conference (ICCC)*, 2013, pp. 38–43.

[30] J. W. Eaton, D. Bateman, and S. Hauberg, *GNU Octave version 3.0.1 manual: a high-level interactive language for numerical computations*. CreateSpace Independent Publishing Platform, 2009, ISBN 1441413006. [Online]. Available: http://www.gnu.org/software/octave/doc/interpreter

[31] I. Emde, W. Berner, and K. Mertens, "Camera systems in hazardous (classified) locations to monitor and control the separation of liquid

waste products in chemical plants," in *Petroleum and Chemical Industry Conference, 2009. PCIC 2009. 2009 Record of Conference Papers - Industry Applications Society 56th Annual*, 2009, pp. 1–7.

[32] V. Feliu-Batlle and F. J. Castillo-García, "On the robust control of stable minimum phase plants with large uncertainty in a time constant. A fractional-order control approach," *Automatica*, vol. 50, no. 1, pp. 218–224, 2014.

[33] C. Festila, E. Dulf, and A. Baldea, "Level and pressure control in condenser of the ^{13}C isotope separation column," in *Automation Quality and Testing Robotics (AQTR), 2010 IEEE International Conference on*, vol. 1, 2010, pp. 1–6.

[34] U. Forssell and L. Ljung, "Closed-loop identification revisited," *Automatica*, vol. 35, no. 7, pp. 1215–1241, 1999.

[35] G. F. Franklin, M. L. Workman, and D. Powell, *Digital Control of Dynamic Systems*, 3rd ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1997.

[36] J.-D. Gabano and T. Poinot, "Fractional modeling applied to non destructive thermal characterization," in *In Proceedings of the 18th IFAC World Congress*, 2011, pp. 13 972–13 977.

[37] A. Gaikwad, P. Vijayan, S. Bhartiya, R. Kumar, H. Lele, and K. Vaze, "Selection of steam drum level control method for multiple drum interacting loops pressure tube-type BWR," *IEEE Transactions on Nuclear Science*, vol. 58, no. 2, pp. 479–489, 2011.

[38] P. E. Gill, W. Murray, and M. H. Wright, *Practical optimization.* London: Academic Press Inc. [Harcourt Brace Jovanovich Publishers], 1981.

[39] H. Gole, P. Barve, A. A. Kesarkar, and N. Selvaganesan, "Investigation of fractional control performance for magnetic levitation experimental set-up," in *Emerging Trends in Science, Engineering and Technology (INCOSET), 2012 International Conference on*, Dec 2012, pp. 500–504.

[40] E. A. Gonzalez, C. A. Monje, L. Dorčák, J. Terpák, and I. Petráš, "A method for incorporating fractional-order dynamics through pid control system retuning," *International Journal of Pure and Applied Mathematics*, vol. 68, no. 4, pp. 593–605, 2013.

[41] M. Halás, "Discrete-time solution to the disturbance decoupling problem of coupled tanks," in *Proc. of the 18th International Conference on Process Control*, M. Fikar and M. Kvasnica, Eds., Tatranská Lomnica, Slovakia, 2011, pp. 162–167.

[42] S. E. Hamamci, "An algorithm for stabilization of fractional-order time delay systems using fractional-order PID controllers." *IEEE Trans. Automat. Contr.*, vol. 52, no. 10, pp. 1964–1969, 2007.

[43] T. Hartley, C. Lorenzo, and H. Qammer, "Chaos in a fractional order Chua's system," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 42, no. 8, pp. 485–490, 1995.

[44] T. T. Hartley and C. F. Lorenzo, "Fractional-order system identification based on continuous order-distributions," *Signal Process.*, vol. 83, pp. 2287–2300, November 2003.

[45] F. Hildebrand, *Introduction to numerical analysis*, ser. International series in pure and applied mathematics. McGraw-Hill, 1956.

[46] R. Hilfer, *Applications of fractional calculus in physics*, ser. Applications of Fractional Calculus in Physics. World Scientific, 2000.

[47] INTECO. (2015) Official website of INTECO, LLC. [Last access time: 12.05.2015]. [Online]. Available: http://www.inteco.com.pl/

[48] C. M. Ionescu, *The Human Respiratory System An Analysis of the Interplay between Anatomy, Structure, Breathing and Fractal Dynamics*. Springer, 2013.

[49] C. Kelley, *Solving Nonlinear Equations with Newton's Method*, ser. Fundamentals of Algorithms. Society for Industrial and Applied Mathematics, 2003.

[50] A. Khadhraoui, K. Jelassi, J.-C. Trigeassou, and P. Melchior, "Identification of fractional model by least-squares method and instrumental variable," *Journal of Computational and Nonlinear Dynamics*, vol. 10, no. 5, pp. 1–10, Sep. 2015.

[51] A. A. Kilbas, H. M. Srivastava, and J. J. Trujillo, *Theory and Applications of Fractional Differential Equations, Volume 204 (North-Holland Mathematics Studies)*. New York, NY, USA: Elsevier Science Inc., 2006.

[52] S. Kuo and B. Lee, *Real-time digital signal processing: implementations, applications, and experiments with the TMS320C55X*. Wiley, 2001.

[53] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright, "Convergence properties of the Nelder-Mead simplex method in low dimensions," *SIAM Journal on Optimization*, vol. 9, no. 1, pp. 112–147, 1998.

[54] C. Lanfeng and X. Dingyu, "Simulation of fractional order control based on ipmc model," in *Control and Decision Conference (2014 CCDC), The 26th Chinese*, May 2014, pp. 598–601.

[55] Y. Lee and J. Watkins, "Determination of all stabilizing fractional-order PID controllers," in *American Control Conference (ACC), 2011*, 2011, pp. 5007–5012.

[56] Y. Lee and J. Watkins, "Determination of all stabilizing fractional-order PID controllers that satisfy a robust performance constraint," in *American Control Conference (ACC), 2013*, 2013, pp. 1789–1794.

[57] L. Ljung, Ed., *System Identification (2nd Ed.): Theory for the User*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 1999.

[58] A. W. Lo, "Long-term memory in stock market prices," *Econometrica*, vol. 59, no. 5, pp. 1279–1313, 1991.

[59] Y. Luo and Y. Chen, "Fractional-order [proportional derivative] controller for robust motion control: Tuning procedure and validation," in *Proc. ACC '09. American Control Conference*, 2009, pp. 1412–1417.

[60] J. A. T. Machado, "Discrete-time fractional-order controllers," *Journal of Fractional Calculus and Applied Analysis*, vol. 4, no. 1, pp. 47–66, 2001.

[61] H. Malek, Y. Luo, and Y. Q. Chen, "Tuning fractional order proportional integral controllers for time delayed systems with a fractional pole," in *Proceedings of the ASME 2011 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, vol. 3, 2011, pp. 1–11.

[62] R. Malti, S. Victor, O. Nicolas, and A. Oustaloup, "System identification using fractional models: State of the art," in *Proc. ASME IDETC/CIE Conference*, Las Vegas, NV, USA, 2007, pp. 295–304.

[63] R. Malti, M. Aoun, J. Sabatier, and A. Oustaloup, "Tutorial on system identification using fractional differentiation models," in *Proc. of 14th IFAC Symposium on System Identification*, Newcastle, Australia, 2006, pp. 606–611.

[64] D. W. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," *SIAM Journal on Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.

[65] I. Mathworks. (2014) Real-Time Windows Target. [Online]. Available: http://www.mathworks.com/products/rtwt/

[66] D. Matignon, "Generalized fractional differential and difference equations: Stability properties and modeling issues," in *Proc. of Math. Theory of Networks and Systems Symposium*, 1998, pp. 503–506.

[67] MATLAB, *version 7.10.0 (R2010a)*. Natick, Massachusetts: The MathWorks Inc., 2010.

[68] Maxim Integrated. (2015) Official website of Maxim Integrated, Inc. [Last access time: 12.05.2015]. [Online]. Available: http://www.maximintegrated.com/

[69] K. Miller and B. Ross, *An introduction to the fractional calculus and fractional differential equations*. Wiley, 1993.

[70] C. A. Monje, B. M. Vinagre, A. J. Calderon, V. Feliu, and Y. Q. Chen, "Auto-tuning of fractional lead-lag compensators," in *Proceedings of the 16th IFAC World Congress*, 2005, pp. 319–324.

[71] C. A. Monje, B. M. Vinagre, V. Feliu, and Y. Q. Chen, "Tuning and auto-tuning of fractional order controllers for industry applications," *Control Engineering Practice*, vol. 16, no. 7, pp. 798–812, 2008.

[72] C. A. Monje, Y. Q. Chen, B. M. Vinagre, D. Xue, and V. Feliu, *Fractional-order Systems and Controls: Fundamentals and Applications*, ser. Advances in Industrial Control. Springer Verlag, 2010.

[73] J. J. Moré and D. C. Sorensen, "Computing a trust region step," *SIAM J. Scientific and Statistical Computing*, vol. 4, pp. 553–572, 1983.

[74] J. J. Moré, "The Levenberg-Marquardt algorithm: Implementation and theory," in *Numerical Analysis*, ser. Lecture Notes in Mathematics, G. Watson, Ed. Springer Berlin Heidelberg, 1978, vol. 630, pp. 105–116.

[75] R. Morrison, "RC constant-argument driving-point admittances," *IRE Transactions on Circuit Theory*, vol. 6, no. 3, pp. 310–317, 1959.

[76] J. Munkres, *Analysis On Manifolds*, ser. Advanced Books Classics Series. Westview Press, 1997.

[77] C. I. Muresan, C. Ionescu, S. Folea, and R. Keyser, "Fractional order control of unstable processes: the magnetic levitation study case," *Nonlinear Dynamics*, pp. 1–12, 2014.

[78] I. Must, F. Kaasik, I. Põldsalu, L. Mihkels, U. Johanson, A. Punning, and A. Aabloo, "Ionic and Capacitive Artificial Muscle for Biomimetic Soft Robotics," *Advanced Engineering Materials*, vol. 17, no. 1, pp. 84–94, 2015.

[79] I. Must, V. Vunder, F. Kaasik, I. P^ oldsalu, U. Johanson, A. Punning, and A. Aabloo, "Ionic liquid-based actuators working in air: The effect of ambient humidity," *Sensors and Actuators B: Chemical*, vol. 202, pp. 114–122, Oct. 2014.

[80] J. A. Nelder and R. Mead, "A simplex method for function minimization," *The Computer Journal*, vol. 7, no. 4, pp. 308–313, 1965.

[81] J. Nocedal, "Theory of algorithms for unconstrained optimization," *Acta numerica*, vol. 1, pp. 199–242, 1992.

[82] Ü. Nurges, "New stability conditions via reflection coefficients of polynomials," *IEEE Transactions on Automatic Control*, vol. 50, no. 9, pp. 1354–1360, 2005.

[83] A. O'Dwyer, *Handbook of PI and PID Controller Tunning Rules*, 3rd ed. Imperial College Press, 2009.

[84] R. Oldenhuis. (2009) Optimize. MathWorks File Exchange. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/24298-optimize

[85] K. Oldham and J. Spanier, *The Fractional Calculus*, ser. Mathematics in science and engineering. Academic Press, 1974.

[86] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Discrete-time signal processing (2nd ed.)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1999.

[87] M. Ortigueira, *Fractional Calculus for Scientists and Engineers*, ser. Lecture Notes in Electrical Engineering. Springer, 2011.

[88] A. Oustaloup, L. L. Lay, and B. Mathieu, "Identification of non integer order system in the time domain," in *Proc. of IEEE Computational Engineering in Systems Application (CESA) Conference*, 1996.

[89] A. Oustaloup, F. Levron, B. Mathieu, and F. M. Nanot, "Frequency-band complex noninteger differentiator: characterization and synthesis," *IEEE Transactions on Circuits and Systems—Part I: Fundamental Theory and Applications*, vol. 47, no. 1, pp. 25–39, 2000.

[90] A. Oustaloup, P. Melchior, P. Lanusse, O. Cois, and F. Dancla, "The CRONE toolbox for Matlab," in *Proc. IEEE Int. Symp. Computer-Aided Control System Design CACSD 2000*, 2000, pp. 190–195.

[91] A. Oustaloup, J. Sabatier, and P. Lanusse, "From fractal robustness to the CRONE control," *Fractional Calculus and Applied Analysis*, vol. 2, no. 1, pp. 1–30, 1999.

[92] F. Padula, R. Vilanova, and A. Visioli, "H_∞ model matching PID design for fractional FOPDT systems," in *American Control Conference (ACC), 2012*, 2012, pp. 5513–5518.

[93] F. Padula and A. Visioli, *Advances in Robust Fractional Control*, ser. SpringerLink : Bücher. Springer International Publishing, 2014.

[94] V. Palmre, D. Brandell, U. Mäeorg, J. Torop, O. Volobujeva, A. Punning, U. Johanson, M. Kruusmaa, and A. Aabloo, "Nanoporous carbon-based electrodes for high strain ionomeric bending actuators," *Smart Materials and Structures*, vol. 18, no. 9, 2009.

[95] I. Petráš, S. Grega, and L. Dorčák, "Digital fractional order controllers realized by PIC microprocessor: Experimental results," in *Proc. of the ICCC'2003 conference*, High Tatras, Slovak Republic, 2003, pp. 873–876.

[96] I. Petráš, *Fractional-Order Nonlinear Systems: Modeling, Analysis and Simulation*, ser. Nonlinear Physical Science. Springer Berlin Heidelberg, 2011.

[97] I. Petráš, "Practical aspects of tuning and implementation of fractional-order controllers," *ASME Conference Proceedings*, pp. 75–84, 2011.

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

[99] I. Podlubny, L. Dorčák, and I. Kostial, "On fractional derivatives, fractional-order dynamic systems and $PI^\lambda D^\mu$-controllers," in *Proc. 36th IEEE Conf. Decision and Control*, vol. 5, 1997, pp. 4985–4990.

[100] I. Podlubny, I. Petráš, B. M. Vinagre, P. O'Leary, and L. Dorčák, "Analogue realizations of fractional-order controllers," *Nonlinear Dynamics*, vol. 29, pp. 281–296, 2002.

[101] I. Podlubny, *Fractional differential equations*, ser. Mathematics in science and engineering. Academic Press, 1999.

[102] D. Pugal, K. Jung, A. Aabloo, and K. J. Kim, "Ionic polymer-metal composite mechanoelectrical transduction: review and perspectives," *Polymer International*, vol. 59, no. 3, pp. 279–289, Mar. 2010.

[103] A. Punning, K. J. Kim, V. Palmre, F. Vidal, C. Plesse, N. Festin, A. Maziz, K. Asaka, T. Sugino, G. Alici, G. Spinks, G. Wallace, I. Must, I. Põldsalu, V. Vunder, R. Temmer, K. Kruusamäe, J. Torop, F. Kaasik, P. Rinne, U. Johanson, A.-L. Peikolainen, T. Tamm, and A. Aabloo, "Ionic electroactive polymer artificial muscles in space applications," *Sci. Rep.*, vol. 4, pp. 1–6, Nov. 2014.

[104] S. Ruiz, B. Mead, V. Palmre, K. J. Kim, and W. Yim, "A cylindrical ionic polymer-metal composite-based robotic catheter platform: modeling, design and control," *Smart Materials and Structures*, vol. 24, no. 1, p. 015007, Jan. 2015.

[105] A. Ruszewski, "Stability regions of closed loop system with time delay inertial plant of fractional order and fractional order PI controller," *Bulletin of the Polish Academy of Sciences, Technical Sciences*, vol. 56, no. 4, pp. 329–332, 2008.

[106] E. Shameli, M. Khamesee, and J. Huissoon, "Nonlinear controller design for a magnetic levitation device," *Microsystem Technologies*, vol. 13, no. 8-10, pp. 831–835, 2007.

[107] D. Simon, *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*. Wiley, 2006.

[108] H. M. Son, "Retuning of PI/PID controllers based on closed-loop model," in *The AUN/SEED-Net Fieldwise Seminar on Control Engineering*, Montien Hotel, Bangkok Thailand, March 2006.

[109] STMicroelectronics. (2015) Official website of STMicroelectronics. [Last access time: 12.05.2015]. [Online]. Available: http://www.st.com/

[110] H. Sun, W. Chen, H. Wei, and Y. Chen, "A comparative study of constant-order and variable-order fractional models in characterizing memory property of systems," *The European Physical Journal Special Topics*, vol. 193, no. 1, pp. 185–192, 2011.

[111] M. Tajjudin, M. H. F. Rahiman, N. M. Arshad, and R. Adnan, "Robust fractional-order PI controller with Ziegler-Nichols rules," *World Academy of Science, Engineering and Technology*, vol. 7, no. 7, pp. 1823–1830, 2013.

[112] A. Tepljakov, E. Petlenkov, and J. Belikov, "FOMCON: a MATLAB toolbox for fractional-order system identification and control," *International Journal of Microelectronics and Computer Science*, vol. 2, no. 2, pp. 51–62, 2011.

[113] A. Tepljakov, E. Petlenkov, and J. Belikov, "FOMCON: Fractional-order modeling and control toolbox for MATLAB," in *Proc. 18th Int. Mixed Design of Integrated Circuits and Systems (MIXDES) Conference*, A. Napieralski, Ed., 2011, pp. 684–689.

[114] A. Tepljakov, E. Petlenkov, and J. Belikov, "A flexible MATLAB tool for optimal fractional-order PID controller design subject to specifications," in *Proceedings of the 31st Chinese Control Conference*, W. Li and Q. Zhao, Eds., Hefei, Anhui, China, 2012, pp. 4698–4703.

[115] A. Tepljakov, E. Petlenkov, and J. Belikov, "Synthesis of digital filters for embedded fractional-order control applications," in *Proc. of the Seventh IKTDK Information and Communication technology Doctoral School Conf.*, 2013, pp. 93–96.

[116] A. Tepljakov, E. Petlenkov, and J. Belikov, "Tuning and digital implementation of a fractional-order PD controller for a position servo," *International Journal of Microelectronics and Computer Science*, vol. 4, no. 3, pp. 116–123, 2013.

[117] A. Tepljakov, E. Petlenkov, and J. Belikov, "Closed-loop identification of fractional-order models using FOMCON toolbox for MATLAB," in *Proc. 14th Biennial Baltic Electronics Conference*, 2014, pp. 213–216.

[118] A. Tepljakov, "Fractional-order calculus based identification and control of linear dynamic systems," Master's thesis, Tallinn University of Technology, 2011.

[119] A. Tepljakov. (2015) Official website of Alpha Control Laboratory. [Last access time: 12.05.2015]. [Online]. Available: http://www.a-lab.ee/

[120] A. Tepljakov, E. Petlenkov, and J. Belikov. (2011) FOMCON toolbox. [Online]. Available: http://www.fomcon.net/

[121] A. Tepljakov, E. Petlenkov, and J. Belikov, "Gain and order scheduled fractional-order PID control of fluid level in a multi-tank system," in *2014 International Conference on Fractional Differentiation and its Applications*, 2014, pp. 1–6.

[122] A. Tepljakov, E. Petlenkov, and J. Belikov, "FOPID controlling tuning for fractional FOPDT plants subject to design specifications in the frequency domain," in *Proc. 2015 European Control Conference (ECC)*, 2015, pp. 3507–3512.

[123] A. Tepljakov, E. Petlenkov, and J. Belikov, "Robust FOPI and FOPID controller design for FFOPDT plants in embedded control applications using frequency-domain analysis," in *Proc. 2015 American Control Conference (ACC)*, 2015, pp. 3868–3873.

[124] A. Tepljakov, E. Petlenkov, J. Belikov, and J. Finajev, "Fractional-order controller design and digital implementation using FOMCON toolbox for MATLAB," in *Proc. of the 2013 IEEE Multi-Conference on Systems and Control conference*, 2013, pp. 340–345.

[125] A. Tepljakov, E. Petlenkov, J. Belikov, and E. A. Gonzalez, "Design of retuning fractional PID controllers for a closed-loop magnetic levitation control system," in *ICARCV 2014 : The 13th International Conference on Control, Automation, Robotics & Vision*, 2014, pp. 1345–1350.

[126] A. Tepljakov, E. Petlenkov, J. Belikov, and M. Halás, "Design and implementation of fractional-order PID controllers for a fluid tank system," in *Proc. 2013 American Control Conference (ACC)*, Washington DC, USA, June 2013, pp. 1780–1785.

[127] J. F. Traub, "Comparison of iterative methods for the calculation of nth roots," *Commun. ACM*, vol. 4, pp. 143–145, March 1961.

[128] J. Trigeassou, T. Poinot, J. Lin, A. Oustaloup, and F. Levron, "Modeling and identification of a non integer order system," in *Control Conference (ECC), 1999 European*, Aug 1999, pp. 2453–2458.

[129] D. Valério and J. Costa, "Levy's identification method extended to commensurate fractional order transfer function," in *EUROMECH Nonlinear Dynamics conference*, Eindhoven, Netherlands, 2005.

[130] D. Valério and J. Costa, "Tuning of fractional PID controllers with Ziegler-Nichols-type rules," *Signal Processing*, vol. 86, no. 10, pp. 2771 – 2784, 2006, special Section: Fractional Calculus Applications in Signals and Systems.

[131] D. Valério and J. Costa, "A review of tuning methods for fractional PIDs," in *Preprints 4th IFAC workshop on fractional differentiation and its applications*, 2010.

[132] D. Valério. (2005) Toolbox ninteger for MatLab, v. 2.3. [Online]. Available: http://web.ist.utl.pt/duarte.valerio/ninteger/ninteger.htm

[133] J. Valsa, P. Dvořá, and M. Friedl, "Network model of the CPE," *Radioengineering*, vol. 20, no. 3, pp. 619–626, September 2011.

[134] A. Varga, "Balancing free square-root algorithm for computing singular perturbation approximations," in *Proc. 30th IEEE Conf. Decision and Control*, 1991, pp. 1062–1065.

[135] M. Čech and M. Schlegel, "The fractional-order PID controller outperforms the classical one," in *Process control 2006*. Pardubice Technical University, 2006, pp. 1–6.

[136] S. Victor, R. Malti, H. Garnier, and A. Oustaloup, "Parameter and differentiation order estimation in fractional models," *Automatica*, vol. 49, no. 4, pp. 926–935, 2013.

[137] B. M. Vinagre, I. Podlubny, A. Hernández, and V. Feliu, "Some approximations of fractional order operators used in control theory and applications," *Fractional Calculus & Applied Analysis*, vol. 3, pp. 945–950, 2000.

[138] B. M. Vinagre, Y. Q. Chen, and I. Petráš, "Two direct Tustin discretization methods for fractional-order differentiator/integrator," *Journal of the Franklin Institute*, vol. 340, no. 5, pp. 349–362, 2003.

[139] V. Vunder, M. Itik, I. Põldsalu, A. Punning, and A. Aabloo, "Inversion-based control of ionic polymer-metal composite actuators with nanoporous carbon-based electrodes," *Smart Materials and Structures*, vol. 23, no. 2, pp. 1–10, 2014.

[140] V. Vunder, M. Itik, A. Punning, and A. Aabloo, "Force control of ionic polymer-metal composite actuators with carbon-based electrodes," in *Proc. SPIE*, vol. 9056, 2014, pp. 1–7.

[141] V. Žilka, M. Halás, and M. Huba, "Nonlinear controllers for a fluid tank system," in *Computer Aided Systems Theory - EUROCAST 2009*, R. Moreno-Díaz, F. Pichler, and A. Quesada-Arencibia, Eds. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 618–625.

[142] H. S. Wall, "A modification of Newton's method," *The American Mathematical Monthly*, vol. 55, no. 2, pp. 90–94, 1948.

[143] J. C. Wang, "Realizations of generalized Warburg impedance with RC ladder networks and transmission lines," *Journal of The Electrochemical Society*, vol. 134, no. 8, pp. 1915–1920, 1987.

[144] M. H. Wright, "Direct Search Methods: Once Scorned, Now Respectable," in *Numerical Analysis 1995 (Proceedings of the 1995*

Dundee Biennial Conference in Numerical Analysis), ser. Pitman Research Notes in Mathematics, D. F. Griffiths and G. A. Watson, Eds., vol. 344. Boca Raton, Florida: CRC Press, 1996, pp. 191–208.

[145] D. Xue and Y. Chen, "A comparative introduction of four fractional order controllers," in *Proceedings of the 4th World Congress on Intelligent Control and Automation*, 2002, pp. 3228–3235.

[146] D. Xue, Y. Q. Chen, and D. P. Atherton, *Linear Feedback Control: Analysis and Design with MATLAB (Advances in Design and Control)*, 1st ed. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2008.

[147] D. Xue, C. Zhao, and Y. Q. Chen, "Fractional order PID control of a DC-motor with elastic shaft: a case study," in *Proc. 2006 American Control Conference (ACC)*, 2006.

[148] J. yi Cao and B.-G. Cao, "Design of fractional order controllers based on particle swarm optimization," in *Industrial Electronics and Applications, 2006 1ST IEEE Conference on*, May 2006, pp. 1–6.

[149] C. Yu, *Autotuning of PID Controllers: A Relay Feedback Approach.* Springer, 2006.

[150] W. Zhang, "An extended ADALINE neural network trained by Levenberg-Marquardt method for system identification of linear systems," in *Control and Decision Conference (CCDC), 2013 25th Chinese*, May 2013, pp. 2453–2458.

[151] C. Zhao, D. Xue, and Y. Q. Chen, "A fractional order PID tuning algorithm for a class of fractional order plants," in *Proc. IEEE Int. Mechatronics and Automation Conf.*, vol. 1, 2005, pp. 216–221.

[152] J. G. Ziegler and N. B. Nichols, "Optimum settings for automatic controllers," in *Transactions of the A.S.M.E.*, 1942, pp. 759–768.

Acknowledgments

The author wishes to express his deep gratitude to his supervisors, Dr. Eduard Petlenkov and Dr. Juri Belikov, for their support and guidance throughout the thesis work; colleague Dr. Kristina Vassiljeva for helpful advice, scientific discussions, as well as for sharing her positive attitude towards solving complicated problems; colleague Dr. Emmanuel Gonzalez for a fruitful scientific collaboration; colleague Sergei Astapov for scientific discussions and his constructive comments related to this work.

The author would also like to thank his family for their strong support.

Finally, the author wishes to thank Associate Professor Miroslav Halás from Slovak University of Technology for the opportunity to hold a number of experiments with the coupled tanks laboratory plant and Professor Alvo Aabloo, head of Intelligent Materials and Systems Laboratory in University of Tartu, for the opportunity to carry out network-based experiments with the IPMC actuator.

This work was partially supported by the Estonian Doctoral School in Information and Communication Technology through the interdisciplinary project FOMCON.

Kokkuvõte

Käesolev väitekiri on pühendatud murruliste tuletiste rakenduste uurimisele dünaamiliste süsteemide identifitseerimisel ja automaatjuhtimissüsteemide disainil. Peamised doktoritöö tulemused on esitatud peatükkides 3–7. Murruliste süsteemide identifitseerimise meetod ajavallas on välja pakutud ning selle abil saadud mudeli analüüs on samuti toodud. Välja pakutud murrulise PID regulaatori disaini meetod põhineb Nelder-Mead simpleksmeetodil ning selle abil saab nõuda kvaliteedinäitajaid nii aja- kui ka sagedusvallas. Võimenduse ja järgu programmeerimismeetod on toodud FOPID reguleerimisahelate jaoks. Stabiliseerimismeetod mittestabiilsete objektide jaoks on kirjeldatud. Klassikaliste PID reguleerimisahelate ümberhäälestamise meetod on välja pakutud, mis võimaldab integreerida murrulist dünaamikat olemasolevasse juhtimisahelasse ilma sisemiste muutusteta. Uuendatud murruliste kondensaatorite aproksimeerimismeetod on kirjeldatud. Efektiivsed analoogfiltrite baasil realiseerimismeetodid on käsitletud. Digitaalfiltrite baasil regulaatorite aproksimeerimismeetod on samuti välja pakutud. Selle põhjal on konstrueeritud reaalne FOPID regulaatori prototüüp, mis on edukalt verifitseeritud reaalaja prototüüpimisplatvormi abil. Peatüki 6 sisuks on FOMCON-i — MATLAB/Simulink keskkonna rakenduspaketi — kirjeldamisele. Meetodid, mis hõlmavad protsessijuhtimise rakendusi, on toodud peatükkide 3 ja 4 lõpus. FFOPDT koos FOPID regulaatoriga juhtimissüsteemi identifitseerimise meetod on välja pakutud, selle süsteemi analüüs on toodud ning FOPID regulaatori häälestamismeetod on kirjeldatud. Peatükis 7 on toodud murruliste regulaatorite reaalsed rakendused. Saadud tulemused näitavad välja pakutavate meetodite efektiivsust, ning demonstreerivad murruliste tuletiste rakenduste eelised.

Abstract

The present thesis is devoted to the research of fractional-order calculus based identification and control of dynamic systems. The main results of the thesis are presented in Chapters 3 through 7. A time domain identification method for fractional-order models is proposed, and the analysis of the quality of the obtained model is provided. Proposed controller design methods include Nelder-Mead simplex method based FOPID controller design subject to specifications in time- and frequency domains. A gain and order scheduling approach is proposed for FOPID control loops. A method for stabilizing unstable plants is proposed leveraging the larger stability zone achieved through use of fractional operators. A conventional PID control loop retuning method is introduced, such that allows incorporating fractional-order dynamics into the existing control loops without internal changes. In the implementation chapter, an updated method for approximating fractional capacitors is introduced, an efficient approach to fractance network generation is proposed, and a digital implementation of FOPID and FOLLC controllers is described. On the basis of this, a hardware FOPID controller prototype is developed and tested using a real-time prototyping platform. Chapter 6 focuses on FOMCON—a software package for the MATLAB/Simulink environment in which most of the results from previous chapters are implemented. Methods applicable to fractional-order process control are found at the end of Chapters 3 and 4, and deal with frequency domain identification of a FFOPDT model based on a relay feedback experiment. The proposed method is verified across a set of FFOPDT models. Complete frequency domain analysis of a control system comprising a FOPID controller and a FFOPDT plant is also provided, and a method is proposed for tuning the FOPID controller. An illustrative example indicates the effectiveness of the method. Finally, applications of fractional-order control are investigated in Chapter 7. The obtained results indicate the effectiveness of the proposed methods, as well as illustrate the benefits that are achieved through the use of fractional calculus based tools.

Elulookirjeldus

1. Isikuandmed

Ees- ja perekonnanimi	Aleksei Tepljakov
Sünniaeg ja -koht	02.06.1987, Tallinn, Eesti
Kodakondsus	Eesti
E-posti aadress	alex@starspirals.net

2. Hariduskäik

Õppeasutus (nimetus lõpetamise ajal)	**Lõpetamise aeg**	**Haridus (eriala/kraad)**
Tallinna Tehnikaülikool	2009	Arvuti- ja süsteemitehnika, B.Sc.
Tallinna Tehnikaülikool	2011	Arvuti- ja süsteemitehnika, M.Sc., Cum Laude

3. Keelteoskus (alg-, kesk- või kõrgtase)

Keel	**Tase**
Eesti	kõrgtase
Inglise	kõrgtase
Vene	emakeel
Prantsuse	algtase
Saksa	algtase

4. Teenistuskäik

Töötamise aeg	**Tööandja nimetus**	**Ametikoht**
2011 – ...	Automaatikainstituut, TTÜ	Insener

5. Projektid

Projekt	**Kirjeldus**
FOMCON	Projekti põhiarendaja

6. Teadustegevus

   Ajakirja- ja konverentsiartiklite loetelu on toodud ingliskeelse CV juures.

7. Patenteeritud leiutised

   A. Tepljakov, E. Petlenkov, and J. Belikov, "Virtuaalne ühendatud anumate süsteem," Eesti patent P201 400 045, 2015, pat. menetluses.

8. Kaitstud lõputööd

   Automaatjuhtimissüsteemide projekteerimine Scilab/Scicos keskkonnas, B.Sc., Tallinna Tehnikaülikool, 2009.

   Murrulistel tuletistel põhinev lineaarsete dünaamiliste süsteemide identifitseerimine ja juhtimine, M.Sc., Tallinna Tehnikaülikool, 2011.

9. Teadustöö põhisuunad

   Murruliste tuletiste teooria, dünaamiliste süsteemide identifitseerimine ja juhtimine, mittelineaarsete juhtimissüsteemide teooria.

Curriculum Vitae

1. Personal Data

Name	Aleksei Tepljakov
Date and place of birth	02.06.1987, Tallinn, Estonia
E-mail address	alex@starspirals.net

2. Education

Educational institution	**Graduation year**	**Education (field of study/degree)**
Tallinn University of Technology	2009	Computer and Systems Engineering, B.Sc.
Tallinn University of Technology	2011	Computer and Systems Engineering, M.Sc., Cum Laude

3. Language competence/skills (fluent, average, basic skills)

Language	**Level**
Estonian	fluent
English	fluent
Russian	native
French	basic skills
German	basic skills

4. Professional employment

Period	**Organization**	**Position**
2011 – ...	Department of Computer Control, TUT	Engineer

5. Projects

Project	**Description**
FOMCON	Main developer

6. Scientific work

1. A. Tepljakov, E. Petlenkov, and J. Belikov, "FOMCON: Fractional-order modeling and control toolbox for MATLAB," in Proc. 18th Int Mixed Design of Integrated Circuits and Systems (MIXDES) Conference, A. Napieralski, Ed., 2011, pp. 684–689.
2. A. Tepljakov, E. Petlenkov, and J. Belikov, "FOMCON: a MATLAB toolbox for fractional-order system identification and control," International Journal of Microelectronics and Computer Science, vol. 2, no. 2, pp. 51–62, 2011.
3. A. Tepljakov, E. Petlenkov, and J. Belikov, "Application of the Newton method to first-order implicit fractional transfer function approximation," in Proc. 19th Int Mixed Design of Integrated Circuits and Systems (MIXDES) Conference, A. Napieralski, Ed., 2012, pp. 473–477.
4. A. Tepljakov, E. Petlenkov, and J. Belikov, "A flexible MATLAB tool for optimal fractional-order PID controller design subject to specifications," in Proceedings of the 31st Chinese Control Conference, W. Li and Q. Zhao, Eds., Hefei, Anhui, China, 2012, pp. 4698–4703.
5. A. Tepljakov, E. Petlenkov, and J. Belikov, "Implementation and real-time simulation of a fractional-order controller using a MATLAB based prototyping platform," in Proc. 13th Biennial Baltic Electronics Conference, 2012, pp. 145–148.
6. A. Tepljakov, E. Petlenkov, and J. Belikov, "Application of Newton's method to analog and digital realization of fractional-order controllers," International Journal of Microelectronics and Computer Science, vol. 3, no. 2, pp. 45–52, 2012.
7. A. Tepljakov, E. Petlenkov, and J. Belikov, "Efficient analog implementations of fractional-order controllers," in Proc. of the 14th International Carpathian Control Conference (ICCC), 2013, pp. 377–382.
8. A. Tepljakov, E. Petlenkov, J. Belikov, and S. Astapov, "Digital fractional-order control of a position servo," in Proc. 20th Int Mixed Design of Integrated Circuits and Systems (MIXDES) Conference, 2013, pp. 462–467.
9. A. Tepljakov, E. Petlenkov, J. Belikov, and M. Halás, "Design and implementation of fractional-order PID controllers for a fluid tank system," in Proc. 2013 American Control Conference (ACC), Washington DC, USA, June 2013, pp. 1780–1785.

10. A. Tepljakov, E. Petlenkov, J. Belikov, and J. Finajev, "Fractional-order controller design and digital implementation using FOMCON toolbox for MATLAB," in Proc. of the 2013 IEEE Multi-Conference on Systems and Control conference, 2013, pp. 340–345.
11. A. Tepljakov, E. Petlenkov, and J. Belikov, "Tuning and digital implementation of a fractional-order PD controller for a position servo," International Journal of Microelectronics and Computer Science, vol. 4, no. 3, pp. 116–123, 2013.
12. A. Tepljakov, E. Petlenkov, and J. Belikov, "Embedded system implementation of digital fractional filter approximations for control applications," in Proc. 21st Int. Mixed Design of Integrated Circuits and Systems (MIXDES) Conference, 2014, pp. 441–445.
13. A. Tepljakov, E. Petlenkov, and J. Belikov, "Gain and order scheduled fractional-order PID control of fluid level in a multi-tank system," in 2014 International Conference on Fractional Differentiation and its Applications, 2014, pp. 1–6.
14. A. Tepljakov, E. Petlenkov, J. Belikov, and E. A. Gonzalez, "Design of retuning fractional PID controllers for a closed-loop magnetic levitation control system," in ICARCV 2014: The 13th International Conference on Control, Automation, Robotics & Vision, 2014, pp. 1345–1350.
15. A. Tepljakov, E. Petlenkov, and J. Belikov, "Closed-loop identification of fractional-order models using FOMCON toolbox for MATLAB," in Proc. 14th Biennial Baltic Electronics Conference, 2014, pp. 213–216.
16. A. Tepljakov, E. Petlenkov, and J. Belikov, "Fractional-order digital filter approximation method for embedded control applications," International Journal of Microelectronics and Computer Science, vol. 5, no. 2, pp. 54–60, 2014.
17. A. Tepljakov, E. Petlenkov, and J. Belikov, "Robust FOPI and FOPID controller design for FFOPDT plants in embedded control applications using frequency-domain analysis," in Proc. 2015 American Control Conference (ACC), 2015, pp. 3868–3873.
18. A. Tepljakov, E. Petlenkov, and J. Belikov, "FOPID controller tuning for fractional FOPDT plants subject to design specifications in the frequency domain," in Proc. 2015 European Control Conference (ECC), 2015, pp. 3507–3512.

7. Patented inventions

   A. Tepljakov, E. Petlenkov, and J. Belikov, "Virtual coupled tank system," Estonian Patent P201 400 045, 2015, pat. pending.

8. Defended theses

   Control System Design in Scilab/Scicos Environment, B.Sc., Tallinn University of Technology, 2009.

   Fractional-order Calculus based Identification and Control of Linear Dynamic Systems, M.Sc., Tallinn University of Technology, 2011.

9. Main areas of scientific work

   Fractional-order calculus, identification and control of dynamic systems, nonlinear control.

Publications

Publication 1

Reference

A. Tepljakov, E. Petlenkov, and J. Belikov, "FOMCON: a MATLAB toolbox for fractional-order system identification and control," International Journal of Microelectronics and Computer Science, vol. 2, no. 2, pp. 51–62, 2011.

Abstract

FOMCON is a new fractional-order modeling and control toolbox for MATLAB. It offers a set of tools for researchers in the field of fractional-order control. In this paper, we present an overview of the toolbox, motivation for its development and relation to other toolboxes devoted to fractional calculus. We discuss all of the major modules of the FOMCON toolbox as well as relevant mathematical concepts. Three modules are presented. The main module is used for fractional-order system analysis. The identification module allows identifying a fractional system from either time or frequency domain data. The control module focuses on fractional-order PID controller design, tuning and optimization, but also has basic support for design of fractional lead-lag compensators and TID controllers. Finally, a Simulink blockset is presented. It allows more sophisticated modeling tasks to be carried out.

FOMCON: a MATLAB Toolbox for Fractional-order System Identification and Control

Aleksei Tepljakov, Eduard Petlenkov, and Juri Belikov

***Abstract*—FOMCON is a new fractional-order modeling and control toolbox for MATLAB. It offers a set of tools for researchers in the field of fractional-order control. In this paper, we present an overview of the toolbox, motivation for its development and relation to other toolboxes devoted to fractional calculus. We discuss all of the major modules of the FOMCON toolbox as well as relevant mathematical concepts. Three modules are presented. The main module is used for fractional-order system analysis. The identification module allows identifying a fractional system from either time or frequency domain data. The control module focuses on fractional-order PID controller design, tuning and optimization, but also has basic support for design of fractional lead-lag compensators and TID controllers. Finally, a Simulink blockset is presented. It allows more sophisticated modeling tasks to be carried out.**

***Index Terms*—fractional calculus, matlab toolbox, automatic control, pid controller, identification, control system design**

I. Introduction

In recent years fractional-order calculus has gained a lot of attention, especially in the field of system theory and control systems design due to more accurate modeling and control enhancement possibilities [1], [2]. Several tools have been developed for fractional order system analysis, modeling and controller synthesis. Among these tools are *MATLAB* toolboxes *CRONE* [3], developed by the CRONE team, and *NINTEGER* [4], developed by Duarte Valério.

The *FOMCON* toolbox for MATLAB [5] is an extension to the mini toolbox introduced in [6], [7], [8], providing graphical user interfaces (GUIs), convenience functions, means of model identification in both time and frequency domains and fractional PID controller design and optimization and a Simulink block set. The goal of the toolbox is to provide an easy-to-use, convenient and useful toolset for a wide range of users. It is especially suitable for beginners in fractional order control because of the availability of GUIs, encompassing nearly every toolbox feature, applied workflow considerations and the ability to get practical results quickly.

This work was supported by the Estonian Doctoral School in Information and Communication Technology under interdisciplinary project FOMCON, the Governmental funding project no. SF0140113As08 and the Estonian Science Foundation Grant no. 8738.

A. Tepljakov is with Department of Computer Control, Tallinn University of Technology Ehitajate tee 5, 19086, Tallinn, Estonia (e-mail: aleksei.tepljakov@dcc.ttu.ee)

E. Petlenkov is with Department of Computer Control, Tallinn University of Technology Ehitajate tee 5, 19086, Tallinn, Estonia (e-mail: eduard.petlenkov@dcc.ttu.ee)

J. Belikov is with Institute of Cybernetics, Tallinn University of Technology, Akadeemia tee 21, 12618, Tallinn, Estonia, e-mail: (e-mail: jbelikov@cc.ioc.ee)

In this paper we present an overview of the FOMCON toolbox and its functions with a summary of used theoretical aspects as well as illustrative examples. The paper is organized as follows. In Section II the reader is introduced to some basic concepts of fractional-order calculus used in control. In Section III an overview of FOMCON toolbox and its features is presented. In Section IV the main module and main GUI facility used for fractional-order system analysis are introduced. Then, the fractional-order identification toolset is presented and discussed in Section V. An overview of the fractional controllers follows in Section VI with particular focus on the $\mathrm{PI}^{\lambda}\mathrm{D}^{\mu}$ control design and optimization. Section VII is devoted to an overview of the provided Simulink blockset which can be used for more sophisticated fractional-order system modeling. In Section VIII some of the current limitations of the toolbox are outlined. Finally, in Section IX conclusions are drawn.

II. An Introduction to Fractional Calculus

Fractional calculus is a generalization of integration and differentiation to non-integer order operator ${}_a\mathscr{D}_t^{\alpha}$, where a and t denote the limits of the operation and α denotes the fractional order such that

$$ {}_a\mathscr{D}_t^{\alpha} = \begin{cases} \frac{\mathrm{d}^{\alpha}}{\mathrm{d}t^{\alpha}} & \Re(\alpha) > 0, \\ 1 & \Re(\alpha) = 0, \\ \int_a^t (\mathrm{d}t)^{-\alpha} & \Re(\alpha) < 0, \end{cases} \tag{1} $$

where generally it is assumed, that $\alpha \in \mathbb{R}$, but it may also be a complex number [7]. There exist multiple definitions of the fractional differintegral. The Riemann-Liouville differintegral is a commonly used definition [8]

$$ {}_a\mathscr{D}_t^{\alpha} f(t) = \frac{1}{\Gamma(m-\alpha)} \left(\frac{\mathrm{d}}{\mathrm{d}\,t}\right)^m \int_a^t \frac{f(\tau)}{(t-\tau)^{\alpha-m+1}}\,\mathrm{d}\tau \tag{2} $$

for $m-1 < \alpha < m$, $m \in \mathbb{N}$, where $\Gamma(\cdot)$ is Euler's gamma function. Consider also the Grünwald-Letnikov definition

$$ {}_a\mathscr{D}_t^{\alpha} f(t) = \lim_{h\to 0} \frac{1}{h^{\alpha}} \sum_{j=0}^{\left[\frac{t-a}{h}\right]} (-1)^j \binom{\alpha}{j} f(t-jh), \tag{3} $$

where $[\cdot]$ denotes the integer part.

The Laplace transform of an α-th derivative with $\alpha \in \mathbb{R}_+$ of a signal $x(t)$ relaxed at $t = 0$ (assuming zero initial conditions) is given by

$$ \mathscr{L}\left\{\mathscr{D}^{\alpha} x(t)\right\} = s^{\alpha} X(s). \tag{4} $$

Copyright © 2011 by Department of Microelectronics & Computer Science, Technical University of Lodz

Thus, a fractional-order differential equation

$$a_n \mathscr{D}^{\alpha_n} y(t) + a_{n-1} \mathscr{D}^{\alpha_{n-1}} y(t) + \cdots + a_0 \mathscr{D}^{\alpha_0} y(t) = b_m \mathscr{D}^{\beta_m} u(t) + b_{m-1} \mathscr{D}^{\beta_{m-1}} u(t) + \cdots + b_0 \mathscr{D}^{\beta_0} u(t), \quad (5)$$

where a_k, $b_k \in \mathbb{R}$ can be expressed as a fractional-order transfer function in form

$$G(s) = \frac{b_m s^{\beta_m} + b_{m-1} s^{\beta_{m-1}} + \cdots + b_0 s^{\beta_0}}{a_n s^{\alpha_n} + a_{n-1} s^{\alpha_{n-1}} + \cdots + a_0 s^{\alpha_0}}. \quad (6)$$

A system given by (6) is said to be of commensurate order if all the orders of the fractional operator s are integer multiples of a base order q such that $\alpha_k, \beta_k = kq$, $q \in \mathbb{R}^+$, $0 < q < 1$. The continuous-time transfer function can be expressed as a pseudo-rational function $H(\lambda)$, where $\lambda = s^q$:

$$H(\lambda) = \frac{\sum_{k=0}^{m} b_k \lambda^k}{\sum_{k=0}^{n} a_k \lambda^k}. \quad (7)$$

Based on this concept, a fractional-order linear time-invariant system can also be represented by a state-space model

$$\begin{aligned} \mathscr{D}^q x(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t). \end{aligned} \quad (8)$$

For more information on fractional-order calculus the interested reader is referred to the books [8], [9], [10], [11].

III. Overview of the FOMCON Toolbox

A. Motivation for Development

FOMCON stands for "Fractional-Order Modeling and Control". The basic motivation for developing it was the desire to obtain a set of useful and convenient tools to facilitate the research of fractional-order systems in application to control system design. This involved writing convenience functions, e.g. the polynomial string parser, and building graphical user interfaces to improve the general workflow. However, a full suite of tools was also desired due to certain limitations in existing toolboxes, which mostly focus on novel control strategies (such as the CRONE control). FOMCON presently aims at extending classical control schemes with concepts of fractional-order calculus. The relation of FOMCON to other MATLAB fractional calculus toolboxes is depicted in Fig. 1.

Further the relation is explained. FOMCON was built upon an existing mini toolbox FOTF. It also uses several functions from NINTEGER toolbox for system identification and if the CRONE toolbox is available, it is also possible to export objects into the CRONE format for further processing. FOMCON also incorporates the `optimize()` function [12]. The latter and the NINTEGER functions are included with respect to the two-clause BSD license.

With all previous considerations, the motivations for developing FOMCON are as follows:

- It is a product suitable for both beginners and more demanding users due to availability of graphical user interfaces and advanced functionality;
- It focuses on extending classical control schemes with concepts of fractional-order calculus;
- It can be viewed as a "missing link" between CRONE and NINTEGER;
- With the Simulink blockset the toolbox aims at a more sophisticated modeling approach;
- The toolbox can be ported to other platforms, such as *Scilab* or *Octave* (some limitations may apply).

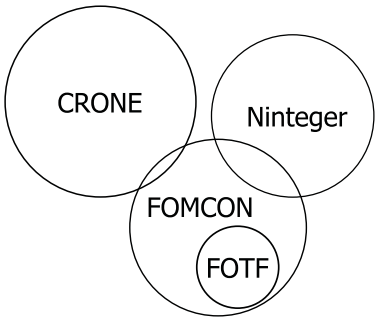


Fig. 1. MATLAB fractional-order calculus toolbox relations to FOMCON

Further we present an overview of the toolbox and its features.

B. Toolbox Features

In FOMCON the main object of analysis is the fractional-order transfer function given by (6). The toolbox focuses on the SISO (single input-single output), LTI (linear time-invariant) systems.

The toolbox is comprised of the following modules:

- Main module (fractional system analysis);
- Identification module (system identification in time and frequency domains);
- Control module (fractional PID controller design, tuning and optimization tools as well as some additional features).

All the modules are interconnected and can be accessed from the main module GUI as depicted in Fig. 2.

A Simulink blockset is also provided in the toolbox allowing complex modeling tasks to be carried out. General approach to block construction is used where applicable.

The FOMCON toolbox relies on the following MATLAB products:

- Control System toolbox, required for most features;
- Optimization toolbox, required for time-domain identification and integer-order PID tuning for common process model approximation.

It is also possible to export fractional-order systems to the CRONE toolbox format (this feature requires the object-oriented CRONE toolbox to be installed).

Further we present an overview of each FOMCON module, providing some theoretical background for the features as well as illustrative examples.

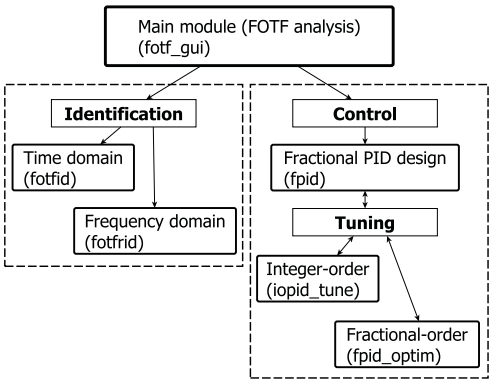


Fig. 2. FOMCON module relations (name of corresponding function to open the GUI is given in parentheses)

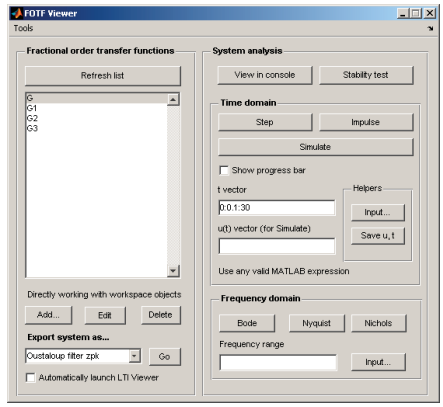


Fig. 3. Main GUI window

IV. Fractional-Order System Analysis

FOMCON provides time-domain and frequency-domain fractional-order system analysis, as well as verifying system stability. In the toolbox fractional-order systems are given by fractional-order transfer function (FOTF) objects in the form (6). These objects are generalizations of the rational transfer functions to the fractional order. To get started one could enter the following into the MATLAB command line

```
fotf_gui
```

The main toolbox GUI called *FOTF Viewer* is then displayed (see Fig. 3). It is divided into two panels:

- The left panel entitled *Fractional order transfer functions* is used to input, edit, delete and convert FOTF objects. The tool is directly working with MATLAB base workspace variables;
- The right panel entitled *System analysis* contains means for fractional-order system analysis in the time domain and in the frequency domain.

The *Tools* menu contains links to the time-domain and frequency-domain identification tools and the fractional PID design tool.

Fractional-order transfer functions may be created in the workspace by pressing the *Add...* button in the GUI. A dialog is shown allowing to enter the zero/pole fractional polynomials of the system (a simple string parser is provided). The system can then be analyzed using the tools in the right panel. Further we discuss the algorithms used to carry out the analysis.

Stability of a fractional-order LTI system (8) can be determined from the following relation

$$\left|\arg\left(\operatorname{eig}\left(A\right)\right)\right| > \gamma\frac{\pi}{2}, \tag{9}$$

where $0 < \gamma < 1$ is the commensurate order of a fractional state-space system and $\operatorname{eig}(A)$ represents the eigenvalues of the associated matrix A. If condition (9) is satisfied, then the system is stable [13]. During the stability test a figure is drawn and populated by the corresponding rational-order system (7) poles. This is an illustration to condition (9): if any of the poles are inside the shaded area of the figure the system is not stable.

Time-domain analysis of the fractional systems, i.e. simulation of the system response to an arbitrary input signal, is carried out using a revised Grünwald-Letnikov definition in (3). The closed-form numerical solution to the fractional-order differential equation is obtained in [7] as

$$y_t = \frac{1}{\sum\limits_{i=0}^{n}\frac{a_i}{h^{\alpha_i}}}\left[u_t - \sum_{i=0}^{n}\frac{a_i}{h^{\alpha_i}}\sum_{j=1}^{\frac{t-a}{h}} w_j^{(\alpha_j)} y_{t-jh}\right], \tag{10}$$

where h is the step-size in computation and $w_j^{(\alpha)}$ can be computed recursively from

$$w_0^{(\alpha)} = 1,\ w_j^{(\alpha)} = \left(1 - \frac{\alpha+1}{j}\right) w_{j-1}^{(\alpha)},\ j = 1, 2, \ldots. \tag{11}$$

The signal $\hat{u}(t)$ is calculated by using (3) substituting $(-1)^{\alpha}\binom{\alpha}{j} = w_j^{(\alpha)}$ and finally the time response under the signal $u(t)$ is obtained. Due to the fixed-step computation the accuracy of the simulation may depend on the chosen step-size h. Thus it is suggested to validate the results by gradually decreasing h until there is no variation in simulation results. Simulation of a large number of points may take a lot of time. A progress bar option is provided to allow keeping track of simulation progress in such cases.

The frequency-domain analysis is done by substituting $s = j\omega$. All the required system frequency characteristics are then obtained using Control System toolbox by supplying the complex frequency response of the plant to the frequency response data object `frd` and then using the standard toolbox frequency response analysis functions `bode()`, `nyquist()`, `nichols()`.

The export facility in the main GUI allows converting FOTF systems into objects of the following type

- Oustaloup filter `zpk`;
- Oustaloup refined filter `zpk`;
- Fractional-order state-space `foss`;
- CRONE `frac_tf`;

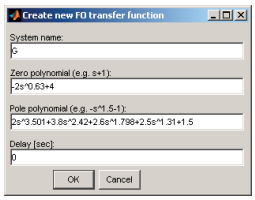


Fig. 4. FOTF entry dialog

- CRONE `frac_ss`.

The last two options are specific to the Object-Oriented CRONE toolbox and require it to be installed.

The Oustaloup filters give a very good approximation of the fractional operators [14] in a specified frequency range (ω_b, ω_h) and of order N. Oustaloup's recursive filter for s^γ for $0 < \gamma < 1$ is given by

$$G_f(s) = K \prod_{k=-N}^{N} \frac{s + \omega'_k}{s + \omega_k}, \tag{12}$$

where ω'_k, ω_k and K are obtained from

$$\begin{aligned} \omega'_k &= \omega_b \left(\frac{\omega_h}{\omega_b}\right)^{\frac{k+N+\frac{1}{2}(1-\gamma)}{2N+1}}, \\ \omega_k &= \omega_b \left(\frac{\omega_h}{\omega_b}\right)^{\frac{k+N+\frac{1}{2}(1+\gamma)}{2N+1}}, \quad K = \omega_h^\gamma. \end{aligned} \tag{13}$$

A refined Oustaloup filter has been proposed in [6]. It is given by

$$s^\alpha \approx \left(\frac{d\omega_h}{b}\right)^\alpha \left(\frac{ds^2 + b\omega_h s}{d(1-\alpha)s^2 + b\omega_h s + d\alpha}\right) G_p, \tag{14}$$

where G_p, ω_k and ω'_k can be computed from

$$\begin{aligned} G_p &= \prod_{k=-N}^{N} \frac{s + \omega'_k}{s + \omega_k}, \\ \omega_k &= \left(\frac{b\omega_h}{d}\right)^{\frac{\alpha+2k}{2N+1}}, \quad \omega'_k = \left(\frac{d\omega_b}{b}\right)^{\frac{\alpha-2k}{2N+1}}. \end{aligned} \tag{15}$$

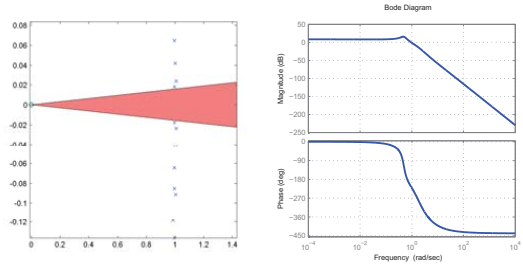
It is expected that a good approximation using (14) is obtained with $b = 10$, $d = 9$.

Fractional-order systems are converted to Oustaloup filter `zpk` objects by approximating fractional orders $\alpha \geq 1$ by $s^\alpha = s^n s^\gamma$, where n denotes the integer part of α and s^γ is obtained by the Oustaloup approximation. Objects exported this way can be analyzed using regular Control System toolbox means. There is also an option to automatically launch the *LTI Viewer* tool upon a successful export.

Example 1. Consider a system given in [7] by

$$G(s) = \frac{-2s^{0.63} + 4}{2s^{3.501} + 3.8s^{2.42} + 2.6s^{1.798} + 2.5s^{1.31} + 1.5}.$$

To supply this system as "G3" one would enter the following in the *Add...* dialog (see Fig. 4).



(a) Stability (b) Bode diagram

Fig. 5. G3 system analysis

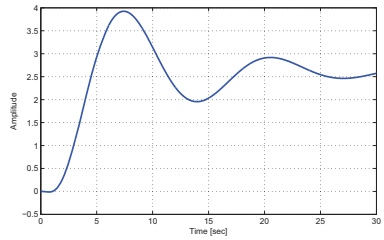


Fig. 6. G3 step response using different calculation methods

Suppose we need to check this system for stability, obtain a Bode diagram and a step response.

The stability analysis illustration is given in Fig. 5a. It can be seen from the zoomed plot that there are no poles inside the shaded region. Therefore, the condition (9) is satisfied and the system is stable. The Bode diagram is shown in Fig. 5b.

A step response in time range $t = [0; 30]$ with a step of $h = 0.01$ is obtained using the Grünwald-Letnikov method (10). The results are given in Fig. 6.

Example 2. Consider a dynamic model of a heating furnace discussed in [15], [16] given by a differential equation

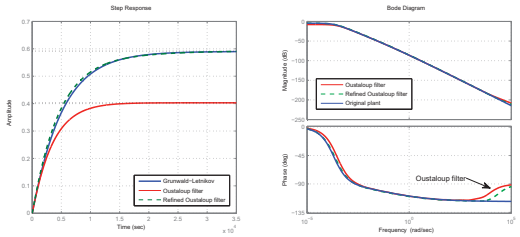
$$a_2 \mathscr{D}^\alpha y(t) + a_1 \mathscr{D}^\beta y(t) + a_0 y(t) = u(t), \tag{16}$$

with $\alpha = 1.31$, $\beta = 0.97$, $a_2 = 14994$, $a_1 = 6009.5$, $a_0 = 1.69$. In the Laplace domain, assuming zero initial conditions, the system is described by a fractional-order transfer function

$$G_1(s) = \frac{1}{14994s^{1.31} + 6009.5s^{0.97} + 1.69}.$$

We shall examine Oustaloup filter approximations of this fractional system. Let us create two filters, an Oustaloup filter `Z1` and a refined Oustaloup filter `Z2` with the default parameters ($\omega = [10^{-4}; 10^4]$, $N = 5$) and compare the resulting system step response (at $t = [0; 35000]$ with $\mathrm{d}t = 0.5$) and frequency response characteristics (Fig. 7a and 7b respectively).

From this example it can be clearly seen that only the refined Oustaloup filter proposed in [6] provides a valid approximation of the fractional-order system than the Oustaloup filter with the same approximation parameters. However, it is also possible to



(a) Step response comparison (b) Frequency response comparison

Fig. 7. Oustaloup and refined Oustaloup filter approximations of system `G1`

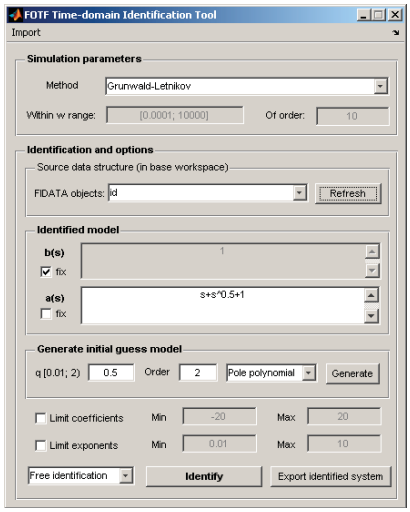


Fig. 8. FOTF Time-domain Identification Tool user interface

obtain a better approximation for this particular system with the Oustaloup filter by shifting the frequency range to $\omega = [10^{-6}; 10^{2}]$.

V. Identification by Fractional-order Model

A. Time-domain Identification

The time-domain identification tool can be accessed from the main GUI via menu *Tools→Identification→Time domain...* or by typing the following into the MATLAB command line:

```
fotfid
```

The corresponding GUI will be launched (depicted in Fig. 8). The tool allows to identify a system by a continuous-time fractional-order model in the form (6).

This is done by fitting an initial model using the least-squares approach minimizing the error norm $\|e(t)\|_2^2$, where

$$e(t) = y(t) - y_{id}(t), \tag{17}$$

by searching for a set of parameters θ of model (6), where

$$\theta = [\ a_p \quad \alpha_p \quad b_z \quad \beta_z\ ] \tag{18}$$

and

$$\begin{aligned} a_p &= [\ a_n \quad a_{n-1} \quad \cdots \quad a_0\ ], \\ \alpha_p &= [\ \alpha_n \quad \alpha_{n-1} \quad \cdots \quad \alpha_0\ ], \\ b_z &= [\ b_m \quad b_{m-1} \quad \cdots \quad b_0\ ], \\ \beta_z &= [\ \beta_n \quad \beta_{n-1} \quad \cdots \quad \beta_0\ ]. \end{aligned} \tag{19}$$

The given parameter set can be further reduced allowing different identification strategies to be applied. The initial model can also be generated form a given commensurate order and the highest order of the model. This is useful when identification is carried out using methods discussed in [17]. The initial guess model can also be imported from the MATLAB workspace.

There is a number of possibilities to fix either fractional-order polynomials, polynomial term coefficients or exponents. Thus a generalized identification tool is obtained, capable of identifying fractional-order systems as well as integer-order systems. There is a possibility to limit the value ranges of the identified parameters. Since it is possible to reduce the number of identified parameters, the identification for complex systems is better conditioned for the underlying optimization task.

A special data structure is used to store the identification data. It can be constructed from the command-line in the following manner:

```
id1 = fidata(y, u, t);
```

where `id1` is the data structure used for identification, `y` is the experimental output signal, `u` is the experimental input signal and `t` is the time vector. The identification tool only works with this type of data structure.

Example 3. Suppose a fractional-order system is given by

$$G_2(s) = \frac{1}{0.8s^{2.2} + 0.5s^{0.9} + 1}.$$

In order to generate identification test data, the following MATLAB commands can be used:

```
t = (0:0.01:20)';
u = zeros(length(t),1);
u(1:200) = ones(200,1);
u(1000:1500) = ones(501,1);
y = lsim(G2,u,t)';
iddata1 = fidata(y,u,t);
```

If nothing but experimental data is known the only way to identify the system is by experimenting with different commensurate orders and orders of the initial model. Also the term coefficients and differentiation orders can be adjusted manually. In this case, if a fractional pole polynomial is selected such, that it is generated with commensurate order $q = 1.2$ and order $N = 2$, the zero polynomial is fixed at $z_p = 1$, free identification is used with term coefficients limited to $c_{lim} = [-20; 20]$ and exponents limited to $e_{lim} = [1 \cdot 10^{-8}; 3]$ then the system is identified as

$$\hat{G}(s) = \frac{1}{0.8s^{2.2} + 0.5s^{0.90001} + s^{9.7153 \cdot 10^{-7}}}.$$

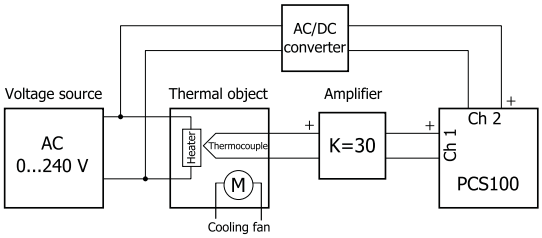


Fig. 9. Experiment schematic diagram

It can be seen that the last term has an order $\alpha = 9.7153 \cdot 10^{-7} \to 0$ and thus the original model is successfully recovered and can be obtained by typing `round(Gid, 1e-4, 1e-4)` in MATLAB. In order to validate the data, one could also type `validate(iddata1, Gid)`. A figure will be drawn showing the comparison of the sampled and identified system responses to the experimental input data as well as the output error.

Example 4. Consider now an example, where a real system is identified. Experimental data is collected from a real thermal object. The schematic diagram corresponding to the experiment is depicted in Fig. 9.

The temperature is measured using a type *K* thermocouple with a DC output of 0...10 mV, amplified with a gain of 30 and fed into a *Velleman PCS100* oscilloscope, which is used to register both the temperature obtained from the amplified thermocouple signal and the voltage source signal. Data was collected from three consecutive experiments. Different voltage set values were used. Due to some limitations of used software and hardware, a total of 1700 points were recorded with a sampling interval of $T_s = 2$ seconds. The obtained system output vector was then filtered ensuring zero phase distortion using a low-pass filter by means of MATLAB `filtfilt()` function, and a transformation was applied so that the output signal vector would contain real temperature values in °C. To account for zero initial conditions requirement the temperature output signal was also shifted such that $t = 0 \to y(t) = 0$. A transformation was applied such that $\hat{u}(t) = 0.01 \cdot u^2(t)$ — the obtained input signal is thus a rough approximation of the final temperature value.

The identification was then carried out using the `fotfid` tool. From previous experience it is known that in case of an integer-order model this system can be approximated by a second order model. Thus, it is possible to obtain the initial guess model by generating a fractional pole polynomial with $q = 1$, $n = 2$ and fixing the zero polynomial at "1" so that a classical, integer order model is initially obtained in the form

$$G_{init}(s) = \frac{1}{s^2 + s + 1}.$$

The free identification method was used with coefficient limits $c_{lim} = [0; 3000]$ and exponent limits $e_{lim} = [10^{-9}; 3]$. The following model was obtained:

$$\hat{G}(s) = \frac{1}{2012.409s^{1.8063} + 107.2882s^{0.93529} + 1.0305}.$$

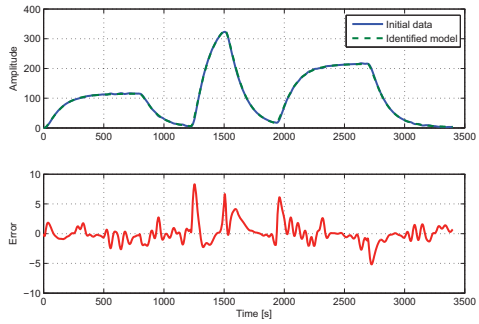


Fig. 10. Thermal system fractional model validation

TABLE I
PROCESS MODEL IDENTIFICATION COMPARISON

	Identified model	Square error norm
$\hat{G}_1(s) =$	$\frac{1}{2012.409s^{1.8063}+107.2882s^{0.93529}+1.0305}$	71.7702
$\hat{G}_2(s) =$	$\frac{0.96039}{2655.4725s^2+151.626s+1}e^{-1.1844s}$	72.3396
$\hat{G}_3(s) =$	$\frac{0.96035}{2835.2438s^2+152.593s+1}$	81.8971

Validation is carried out with the same dataset as for identification. The corresponding plot is given in Fig. 10.

Two integer-order models were obtained from the same experimental dataset by using the MATLAB Identification toolbox for comparison. Results are provided in Table I.

Taking the square error norm as a measure of model precision, one could say that the fractional-order model $\hat{G}_1$ is more accurate than the integer-order models $\hat{G}_2$ and $\hat{G}_3$. This is to be expected due to the properties of fractional operators and the extra degrees of modeling freedom and allows for an improvement in control system characteristics. However, in order to claim this explicitly one would need to use hardware and software methods with a more strict precision requirement.

B. Frequency-domain Identification

The time-domain identification tool can be launched from *FOTF Viewer* via menu *Tools→Identification→Frequency domain...* or by typing the following into the MATLAB command line:

```
fotfrid
```

The frequency-domain identification tool GUI will then be displayed. The tool allows to identify a fractional-order model either in the form

$$G(s) = \frac{1}{c_n s^{n\gamma} + c_{n-1} s^{(n-1)\gamma} + \cdots + c_1 s^{\gamma} + c_0} \tag{20}$$

or in the form

$$G(s) = \frac{b_m s^{m\gamma} + b_{m-1} s^{(m-1)\gamma} + \cdots + b_1 s^{\gamma} + b_0}{a_n s^{n\gamma} + a_{n-1} s^{(n-1)\gamma} + \cdots + a_1 s^{\gamma} + 1}, \tag{21}$$

where γ is the fractional-order transfer function commensurate order and n, m are the corresponding polynomial orders. Further we discuss the identification methods used by the tool.

Three algorithms for system identification in the frequency domain are available [18], [19]. The Hartley method allows to obtain the model (20) parameters $c_0, c_1, \ldots, c_n$ from the experimentally collected complex frequency response by solving the following equation

$$\begin{bmatrix} \frac{1}{G(j\omega_1)} \\ \frac{1}{G(j\omega_2)} \\ \vdots \\ \frac{1}{G(j\omega_m)} \end{bmatrix} = \tag{22}$$

$$\begin{bmatrix} 1 & (j\omega_1)^{\gamma} & (j\omega_1)^{2\gamma} & \cdots & (j\omega_1)^{n\gamma} \\ 1 & (j\omega_2)^{\gamma} & (j\omega_2)^{2\gamma} & \cdots & (j\omega_2)^{n\gamma} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & (j\omega_m)^{\gamma} & (j\omega_m)^{2\gamma} & \cdots & (j\omega_m)^{n\gamma} \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix},$$

where $\omega_1, \omega_2, \ldots, \omega_n$ are the sampling frequencies. When using this algorithm, the user must supply the commensurate order γ as well as model order n.

The Levy and Vinagre identification methods allow to identify a fractional-order model in the form (21). The underlying algorithm for both methods is the same. It finds the parameters for an experimental frequency response given by $G(j\omega) = \Re(\omega) + j\Im(\omega)$ by solving the following equation

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} b_0 \\ \vdots \\ b_m \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} e \\ g \end{bmatrix}, \tag{23}$$

where $b_0, \ldots b_m$ and $a_1, \ldots a_n$ are the identified model parameters and A, B, C, D and e, g are constructed from the experimental data. Please see [19] for detailed explanation of these parameters.

During identification using Levy's method the following square norm is minimized:

$$\begin{aligned} \epsilon &= G(j\omega)\left[a_n(j\omega)^{n\gamma} + \cdots + a_1(j\omega)^{\gamma} + 1\right] \\ &- \left[b_m(j\omega)^{m\gamma} + \cdots + b_1(j\omega)^{\gamma} + b_0\right]. \end{aligned} \tag{24}$$

The Vinagre method adds weights to the norm in order to improve the approximation at low frequencies such that $\epsilon' = w \cdot \epsilon$, where weights w are frequency dependent and for frequencies ω_i, $i = 1, ..., f$ they are given by

$$w = \begin{cases} \frac{\omega_2 - \omega_1}{2\omega_1^2} & , \quad i = 1, \\ \frac{\omega_{i+1} - \omega_{i-1}}{2\omega_i^2} & , \quad 1 < i < f, \\ \frac{\omega_f - \omega_{f-1}}{2\omega_f^2} & , \quad i = f. \end{cases}$$

In order to use these methods, the user needs to supply the commensurate order, as well as fractional polynomial orders n and m. This allows for an additional optimization problem to be stated for a set of parameters $\theta = \begin{bmatrix} \gamma & n & m \end{bmatrix}$. An objective function to minimize is given by a performance index in the form

$$J = \frac{1}{n_\omega} \sum_{i=1}^{n_\omega} \left| G(j\omega) - \hat{G}(j\omega) \right|^2,$$

where n_ω is the number of frequencies in ω, G is the original plant, from which the response was obtained, and $\hat{G}$ is the identified plant. The error index is also used to evaluate the identification result in general.

As with the time-domain identification, a special data structure `ffidata` is used to hold the experimental frequency response. It can be constructed from the MATLAB command-line as follows:

```
id1 = ffidata(mag, ph, w);
```

or

```
id1 = ffidata(r, w);
```

where `mag` is the observed frequency response magnitude in dB, `ph` is the observed frequency response phase angle in degrees and `w` is the vector containing frequencies in rad/s, where the response is known. Alternatively, it is possible to create the identification data structure using the complex response `r`.

Example 5. In this example we will illustrate the use of the frequency-domain identification tool. Consider a plant given by

$$G(s) = \frac{s^{0.32} + 5}{100s^{1.92} + 20s^{0.96} - 5s^{0.64} + 1}.$$

Let us generate an identification dataset `fid1` with 50 logarithmically spaced frequency sample points in the range $\omega = [10^{-4}; 10^{4}]$. This can be done by writing the following into the MATLAB command line:

```
G    = newfotf('s^0.32+5', ...
       '100s^1.92+20s^0.96-5s^0.64+1');
w    = logspace(-4, 4, 50);
fid1 = ffidata(freqresp(G, w), w);
```

We will now identify the fractional model by means of the `fotfrid` tool.

As with the time-domain identification, if no information about the model is given, the only way to identify it is by trial and error. The good news is that frequency-domain identification is fast. In this case, one could use the Vinagre method and choose a commensurate order $q = 0.2$, leaving the polynomial orders at their default value $n = m = 5$, and also apply the best fit identification by going to *Tools→Best fit* and setting the maximum model orders to $N = 6$ in the optimization settings dialog. With these settings applied the

following model is obtained:

$$\begin{aligned}\hat{G}(s) &= \frac{\hat{b}(s)}{\hat{a}(s)},\\ \hat{b}(s) &= 2.6322\cdot 10^{-15}s^{1.92} - 1.4416\cdot 10^{-13}s^{1.6}\\ &+ 3.2699\cdot 10^{-12}s^{1.28} - 3.7288\cdot 10^{-11}s^{0.96}\\ &+ 2.1969\cdot 10^{-10}s^{0.64} + s^{0.32} + 5,\\ \hat{a}(s) &= 100s^{1.92} + 1.6987\cdot 10^{-8}s^{1.6}\\ &- 1.0219\cdot 10^{-8}s^{1.28} + 20s^{0.96} - 5s^{0.64}\\ &- 1.5758\cdot 10^{-10}s^{0.32} + 1.\end{aligned}$$

The initial system can be obtained using the following:

```
G = trunc(Gid, 1e-5, 1e-5)
```

Obviously, the system used in this example was relatively easy to identify. In practical cases, one should carefully consider the choice of the commensurate order. The identified system polynomials may be of a very high order, in which case the *Best fit* tool will be less useful due to large computational efforts involved.

For a practical example of frequency-domain identification see the fractional lead-lag compensator realization in Example 7.

VI. Fractional-order Control

In this section we discuss the fractional-order controllers used in the FOMCON toolbox, namely the fractional PID controller, fractional lead-lag compensator and the TID controller. Our main focus will be on the fractional PID controller, due to its importance in the industry [7].

A. $PI^{\lambda}D^{\mu}$ Controller Design, Tuning and Optimization

The fractional-order PID controller was first introduced by Podlubny in [1]. This generalized controller is called the $PI^{\lambda}D^{\mu}$ controller (notation $PI^{\lambda}D^{\delta}$ is also used in literature) and has an integrator with an order λ and a differentiator of order μ. Recent researches show that the fractional-order PID outperforms the classical PID [20], [21].

The fractional PID controller transfer function has the following form

$$G_c(s) = K_p + \frac{K_i}{s^{\lambda}} + K_d s^{\mu}. \tag{25}$$

Obviously, when taking $\lambda = \mu = 1$ the result is the classical integer-order PID controller. With more freedom in tuning the controller, the four-point PID diagram can now be seen as a PID controller plane, which is conveyed in Fig. 11.

The fractional-order PID design tool can be accessed from the main GUI by *Tools→Fractional PID design* or by the `fpid` command. It allows to design a $PI^{\lambda}D^{\mu}$ controller for a typical negative feedback unity system shown in Fig. 12.

There are several approaches for fractional PID design which depend on the plant to be controlled. If the plant is given by an integer-order model, then classical tuning procedures could be employed to obtain integer-order PID parameters.

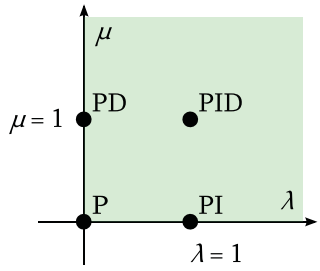


Fig. 11. The $PI^{\lambda}D^{\mu}$ controller plane

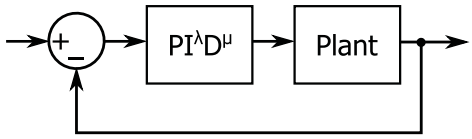


Fig. 12. Feedback control system with fractional PID controller

Fractional PID orders can then be tuned to achieve enhanced performance. A tool is provided which permits identifying the process (which could also be fractional-order) by well-known models (FOPDT, IPDT, FOIPDT) and computing integer-order PID gains using classical tuning strategies such as Ziegler-Nichols, Åström-Hägglund etc. [6]. This tool can be accessed from the menu *Tuning→Integer-order PID* or by typing `iopid_tune`. For fractional-order PID tuning consider methods proposed in [8], [22].

Another case is when the plant is of fractional-order. No special tuning method is currently provided. However, a tuning method for a class of plants can be found in [16].

The optimization tool, provided in FOMCON, can in practice be used for fractional PID tuning due to its flexibility. The tool can be accessed from the PID design tool menu *Tuning→Optimize* (or by typing `fpid_optim`). The tool is shown in Fig. 13. Here is a summary of the options provided:

- Plant model and fractional PID approximation type. Only Oustaloup filter type simulations are used mainly due to processing speed.
- Possibility to tune all parameters, fix gains or fix fractional exponents.
- Possibility to constrain every tuned parameter, except for the lower bound of the exponents which is fixed.
- Optimization to several performance metrics (ISE, IAE, ITSE, ITAE).
- Some control over control system performance specifications (gain and phase margin).
- User-defined number of optimization iterations.

The optimization tool uses the `optimize` function [12] in order to tune the fractional PID parameters by minimizing the function given by the corresponding performance index. These are as follows:

- Integral square error $ISE = \int_0^t e^2(t)\ \mathrm{d}t$,
- Integral absolute error $IAE = \int_0^t |e(t)|\ \mathrm{d}t$,
- Integral time-square error $ITSE = \int_0^t te(t)^2\ \mathrm{d}t$,

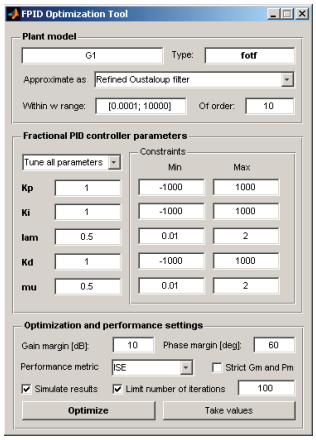


Fig. 13. Fractional PID optimize tool

- Integral time-absolute error $ITAE = \int_0^t t\,|e(t)|\,\mathrm{d}t$,

where $e(t) = 1 - y(t)$, $y(t)$ is the tuned fractional control system step response.

Example 6. Consider the model of a thermal object we obtained through identification in Example 4:

$$G = \frac{1}{2012.4087s^{1.8063} + 107.2882s^{0.93529} + 1.0305}.$$

We will now design a PI$^{\lambda}$D$^{\mu}$ controller for this plant using the optimization tool.

Initially the PID parameters are set to $K_p = K_i = K_d = 100$, $\lambda = \mu = 1$. The exponents are fixed so that an integer-order PID could be designed. Search limits are set to $K = [-500; 500]$ for gains and $\gamma = [0.01; 2]$. For simulation, the refined Oustaloup filter approximation is used with default parameters ($\omega = [0.0001; 10000]$, $N = 10$). Specifications are as follows. Gain margin is set to 10 dB, while phase margin to 45 degrees (non-strict). Performance metric is *IAE*.

Optimization with these settings leads to the following integer-order PID controller parameter set: $K_p = 457.8607$, $K_i = 0.97807$, $K_d = 408.3947$. Obtained open-loop phase margin is $\varphi_m = 45.01°$. Next the gains are fixed and integrator and differentiator orders are set to $\lambda = \mu = 0.5$. The strict option is enabled. The optimization is then continued. As a result, the orders are found such that $\lambda = 0.24726$ and $\mu = 0.7528$.

A comparison of simulation of the designed control systems with a set value $SV = 150$ is shown in Fig. 14. It can be seen, that by tuning only the orders of the controller a better result is achieved. It is important to note, however, that this result is obtained with an unconstrained control effort value, which is always limited in practical situations. Thus it may be required to review the controller settings according to these limitations for practical use.

B. Fractional Lead-Lag Compensator

Lead-lag compensators are a well-known type of feedback controller widely used in practice. Extending it with ideas from fractional calculus can lead to a more robust controller.

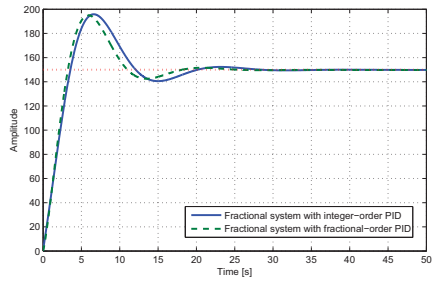


Fig. 14. Control system for thermal object comparison

A fractional-order lead-lag compensator has the following transfer function:

$$G_c(s) = k' \left(\frac{\lambda s + 1}{x\lambda s + 1} \right)^{\alpha}, \tag{26}$$

where α is the fractional order of the controller, λ and x are parameters such that $\frac{1}{\lambda} = \omega_z$ is the zero frequency and $\frac{1}{x\lambda} = \omega_p$ is the pole frequency and $k' = K_c x^{\alpha}$. When $\alpha > 0$ the controller (26) corresponds to a fractional-order lead compensator and when $\alpha < 0$ it corresponds to a fractional lag compensator.

The contribution of parameter α is such, that the lower its value, the longer the distance between the zero and pole and vice versa so that the contribution of phase at a certain frequency stands still. This makes the controller more flexible and allows a more robust approach to the design. Tuning and auto-tuning techniques are discussed in [8].

No specialized tool is yet available in FOMCON for fractional lead-lag controller tuning. However, tools are proposed for the analysis of this controller. Since the controller is given in implicit form, to obtain a transfer function the following can be done:

- Obtain a complex frequency response of the controller, a special function `frlc()` is available in FOMCON for this task;
- Identify the controller using an appropriate tool.

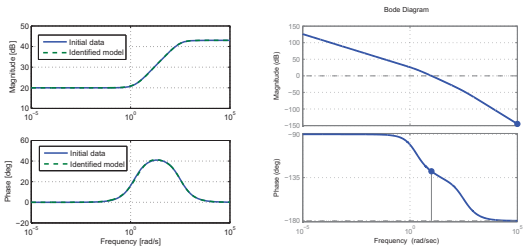
Further we illustrate this procedure.

Example 7. Consider an integer-order plant given by a model

$$G(s) = \frac{2}{s(0.5s + 1)}.$$

In this example, we will realize a fractional lead-lag compensator for this plant discussed in [8]. The gain crossover frequency is chosen such that $\omega_{cg} = 10\,rad/sec$. At this frequency the plant has a magnitude of $-28.1291\,\mathrm{dB}$ and a phase of $-168.69°$. To achieve a magnitude of 0 dB at the gain crossover frequency and a phase margin $\varphi_m = 50°$ the fractional lead compensator is designed with parameters $k' = 10$, $x = 0.005$, $\lambda = 0.6404$, $\alpha = 0.5$ and thus has the following implicit fractional-order transfer function

$$G_c(s) = 10 \left(\frac{0.6404s + 1}{0.0032s + 1} \right)^{0.5}.$$



(a) Controller identification result (b) Control system open-loop frequency response

Fig. 15. Fractional lead compensator realization

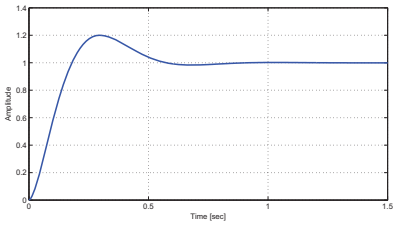


Fig. 16. Control system with fractional lead compensator step response

In order to implement this controller, the frequency response data is obtained using the `frlc()` function in the range $\omega = [10^{-5}; 10^{5}]$ and a frequency-domain identification dataset is created by typing the following in MATLAB:

```
w   = logspace(-5,5,1000);
r   = frlc(10,0.005,0.6404,0.5,w);
flc = ffidata(r,w);
```

Next, the `fotfrid` tool is used to obtain a fractional-order approximation of the compensator. With $q = 0.492$, setting both polynomial orders to 4 and using the Vinagre method, the following fractional-order transfer function is obtained with an error $J = 0.014867$:

$$\hat{G}_c(s) = \frac{\hat{b}(s)}{\hat{a}(s)},$$

$$\hat{b}(s) = 0.031325s^{1.968} + 0.30643s^{1.476} + 4.6284s^{0.984} + 4.0234s^{0.492} + 10.0005,$$

$$\hat{a}(s) = 0.0002215s^{1.968} + 0.0021625s^{1.476} + 0.061928s^{0.984} + 0.41302s^{0.492} + 1.$$

The frequency fitting result is also shown in Fig. 15a. The open-loop control system frequency response is given in Fig. 15b. It can be seen, that the desired crossover frequency $\omega_{cg} = 9.94$ and phase margin $\varphi_m = 51.6^\circ$ are very close to specification.

Finally, the step response of the designed control system is given in Fig. 16.

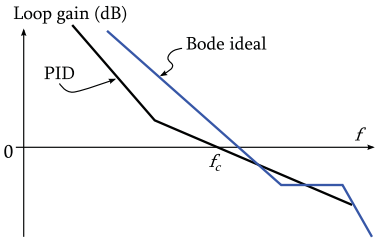


Fig. 17. Bode plots for PID controlled plant and the ideal loop response

C. TID Controller

The TID (tilt-integral-derivative) controller was first proposed in [23]. The structure of the TID controller is given by the following transfer function

$$G_c(s) = \frac{K_t}{s^{\frac{1}{n}}} + \frac{K_i}{s} + K_d s, \tag{27}$$

where $K_t/s^{\frac{1}{n}}$ is the *Tilt* type compensator and $n \in \mathbb{R}$, $n > 0$, preferably $n \in [2; 3]$. It can be seen, that the TID controller corresponds to a conventional PID controller with proportional gain replaced by the compensator component. The motivation for this type of controller is from the consideration of Bode's theoretically optimal loop response (see Fig. 17). A possible tuning strategy according to this consideration is given in [23], [24].

In order to obtain a fractional transfer function in FOMCON, one could use the following:

```
Gc = tid(Kt, n, Ki, Kd);
```

where parameters K_t, n, K_i, K_d correspond to those in (27).

VII. Modeling in Simulink

The FOMCON Simulink block library currently consists of eight blocks and is shown in Fig. 18.

The library is based on Oustaloup filter approximation by means of the `oustapp()` function. The discrete blocks use the Control System toolbox function `c2d()` to obtain the discrete model from the Oustaloup filter LTI system. General block structure is used where applicable.

The difference between the *Fractional operator* and *Fractional derivative* blocks is that the order α of the former is limited to $0 < \alpha < 1$.

In order to ensure efficient and accurate simulation, the model built with these blocks may be made up of stiff systems and an appropriate solver should be used in Simulink in such a case (`ode15s` or `ode23tb`).

Example 8. Consider a model of a dynamic system and the corresponding fractional-order controller discussed in [6] and given by the following transfer functions:

$$G(s) = \frac{1}{0.8s^{2.2} + 0.5s^{0.9} + 1},$$

$$G_c(s) = 233.4234 + \frac{22.3972}{s^{0.1}} + 18.5274 \cdot s^{1.15}.$$

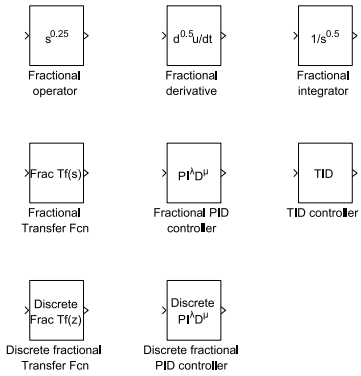


Fig. 18. FOMCON Simulink library

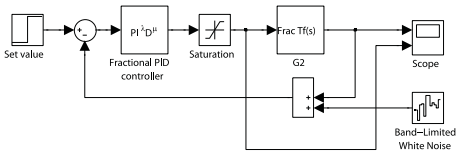


Fig. 19. System model in Simulink

Let us build the corresponding model in Simulink, using the above blockset. The resulting model is given in Fig. 19. A saturation block is added, limiting the control signal within an interval $U_{lim} = [-100; 100]$ and adding a band-limited white noise block for simulating disturbance in the system with power of $P = 10^{-9}$, sample time of $T = 0.01$ and seed value of 23341. System simulation result is given in Fig. 20.

VIII. Discussion

The FOMCON toolbox was developed and tested in MATLAB v. 7.7. However, most of the features are backwards-compatible and were tested with earlier releases of MATLAB (versions 7.4-7.6). FOMCON requires System Control toolbox for general functionality and Optimization toolbox for model identification in the time domain.

Further, we discuss some of the current limitations of the FOMCON toolbox.

- The PID optimization tool lacks complete control over control system gain and phase margins. The algorithm can only guarantee that the minimum given specifications are met by evaluating the open-loop control system at every optimization step when the *Strict* option is checked. However, the initial fractional PID parameters should strictly satisfy the minimum specifications or else optimization will not be carried out and an error will be issued.
- More design specifications settings are required for PID tuning, including minimum and maximum allowed value settings for the control effort.
- Both the identification and optimization tools work with numbers at a fixed accuracy of four decimal places.
- Time domain identification tool does not yet identify the system lag parameter.
- There are no automatic tuning algorithms implemented for the fractional lead-lag compensator and TID controller.
- While the order of the fractional derivative block in Simulink can have an order $\alpha > 1$, the accuracy of the simulation will be reduced with higher orders.

The current limitations of the FOMCON toolbox will be the subject of further development and will be gradually eliminated in future releases.

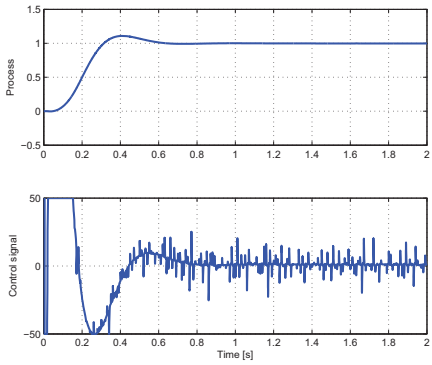


Fig. 20. Fractional-order control system simulation result

IX. Conclusions

In this paper, we presented a MATLAB toolbox containing the necessary tools to work with a class of fractional-order models in control. Theoretical aspects behind the tools were also covered with illustrative examples. A set of graphical user interfaces was introduced with relevant comments. We have discussed fractional-order system analysis, identification in both time and frequency domains and a set of fractional-order controllers, focusing on tuning and optimization of the fractional PID controller. The performance of the latter was found to be superior to an integer-order PID obtained during the same tuning procedure. A Simulink blockset was also presented in the paper.

References

[1] I. Podlubny, "Fractional-order systems and PI$^{\lambda}$D$^{\mu}$-controllers," vol. 44, no. 1, pp. 208–214, 1999.
[2] M. Yahyazadeh and M. Haeri, "Application of fractional derivative in control functions," in *Proc. Annual IEEE India Conf. INDICON 2008*, vol. 1, 2008, pp. 252–257.
[3] A. Oustaloup, P. Melchior, P. Lanusse, O. Cois, and F. Dancla, "The CRONE toolbox for Matlab," in *Proc. IEEE Int. Symp. Computer-Aided Control System Design CACSD 2000*, 2000, pp. 190–195.
[4] D. Valrio. (2005) Toolbox ninteger for MatLab, v. 2.3. [Online]. Available: http://web.ist.utl.pt/duarte.valerio/ninteger/ninteger.htm
[5] A. Tepljakov, E. Petlenkov, and J. Belikov. (2011) FOMCON toolbox. [Online]. Available: http://www.fomcon.net/

[6] D. Xue, Y. Chen, and D. P. Atherton, *Linear Feedback Control: Analysis and Design with MATLAB (Advances in Design and Control)*, 1st ed. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2008.
[7] Y. Q. Chen, I. Petras, and D. Xue, "Fractional order control - a tutorial," in *Proc. ACC '09. American Control Conference*, 2009, pp. 1397–1411.
[8] C. A. Monje, Y. Chen, B. Vinagre, D. Xue, and V. Feliu, *Fractional-order Systems and Controls: Fundamentals and Applications*, ser. Advances in Industrial Control. Springer Verlag, 2010.
[9] K. Miller and B. Ross, *An introduction to the fractional calculus and fractional differential equations*. Wiley, 1993.
[10] I. Podlubny, *Fractional differential equations*, ser. Mathematics in science and engineering. Academic Press, 1999.
[11] A. A. Kilbas, H. M. Srivastava, and J. J. Trujillo, *Theory and Applications of Fractional Differential Equations, Volume 204 (North-Holland Mathematics Studies)*. New York, NY, USA: Elsevier Science Inc., 2006.
[12] R. Oldenhuis. (2009) Optimize. MathWorks File Exchange. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/24298-optimize
[13] D. Matignon, "Generalized fractional differential and difference equations: Stability properties and modeling issues," in *Proc. of Math. Theory of Networks and Systems Symposium*, 1998, pp. 503–506.
[14] A. Oustaloup, F. Levron, B. Mathieu, and F. M. Nanot, "Frequency-band complex noninteger differentiator: characterization and synthesis," vol. 47, no. 1, pp. 25–39, 2000.
[15] I. Podlubny, L. Dorcak, and I. Kostial, "On fractional derivatives, fractional-order dynamic systems and $PI^{\lambda}D^{\mu}$-controllers," in *Proc. 36th IEEE Conf. Decision and Control*, vol. 5, 1997, pp. 4985–4990.
[16] C. Zhao, D. Xue, and Y. Chen, "A fractional order pid tuning algorithm for a class of fractional order plants," in *Proc. IEEE Int Mechatronics and Automation Conf*, vol. 1, 2005, pp. 216–221.
[17] A. Oustaloup, R. Malti, M. Aoun, and J. Sabatier, "Tutorial on system identification using fractional differentiation models," in *14th IFAC Symposium on System Identification*, 2006, pp. 606–611.
[18] T. T. Hartley and C. F. Lorenzo, "Fractional-order system identification based on continuous order-distributions," *Signal Process.*, vol. 83, pp. 2287–2300, November 2003.
[19] D. Valrio and J. Costa, "Levy's identification method extended to commensurate fractional order transfer function," in *EUROMECH Nonlinear Dynamics conference*, Eindhoven, Netherlands, 2005.
[20] Y. Luo and Y. Chen, "Fractional-order [proportional derivative] controller for robust motion control: Tuning procedure and validation," in *Proc. ACC '09. American Control Conference*, 2009, pp. 1412–1417.
[21] M. Čech and M. Schlegel, "The fractional-order pid controller outperforms the classical one," in *Process control 2006*. Pardubice Technical University, 2006, pp. 1–6.
[22] C. Yeroglu, C. Onat, and N. Tan, "A new tuning method for $PI^{\lambda}D^{\mu}$ controller," in *Proc. Int. Conf. Electrical and Electronics Engineering ELECO 2009*, vol. II, 2009, pp. 312–316.
[23] B. J. Lurie, "Three-parameter tunable tilt-integral-derivative (TID) controller," US Patent US5 371 670, 1994.
[24] D. Xue and Y. Chen, "A comparative introduction of four fractional order controllers," in *Proceedings of the 4th World Congress on Intelligent Control and Automation*, 2002, pp. 3228–3235.

Aleksei Tepljakov was born in 1987 in Tallinn. He received his B.Sc and M.Sc in computer and systems engineering from Tallinn University of Technology. He is currently working at the Department of Comupter Control at Tallinn University of Technology. His main research interests include fractional-order control of complex systems and fractional filter based signal processing.

Eduard Petlenkov Eduard Petlenkov was born in 1979. He recieved his B.Sc, M.Sc and PhD degrees in computer and systems engineering from Tallinn University of Technology. He is an Associate Professor in the Department of Computer Control at Tallinn University of Technology. His main research interests lie in the domain of nonlinear control, system analysis and computational intelligence.

Juri Belikov was born in 1985. He recieved his B.Sc degree in mathematics from Tallinn University, and his M.Sc in computer and systems engineering from Tallinn University of Technology. He joined the the Institute of Cybernetics and Department of Computer Control at Tallinn University of Technology. His main research interests lie in the domain of nonlinear control theory.

Publication 2

Reference

A. Tepljakov, E. Petlenkov, J. Belikov, and M. Halás, "Design and implementation of fractional-order PID controllers for a fluid tank system," in Proc. 2013 American Control Conference (ACC), Washington DC, USA, June 2013, pp. 1780–1785.

Abstract

In this paper, we investigate the practical problems of design and digital implementation of fractional-order PID controllers used for fluid level control in a system of coupled tanks. We present a method for obtaining the $PI^{\lambda}D^{\mu}$ controller parameters and describe the steps necessary to obtain a digital implementation of the controller. A real laboratory plant is used for the experiments, and a hardware realization of the controller fit for use in embedded applications is proposed and studied. The majority of tasks is carried out by means of the FOMCON ("Fractional-order Modeling and Control") toolbox running in the MATLAB computing environment.

2013 American Control Conference (ACC)
Washington, DC, USA, June 17-19, 2013

Design and Implementation of Fractional-order PID Controllers for a Fluid Tank System

Aleksei Tepljakov[1], Eduard Petlenkov[1], Juri Belikov[2], and Miroslav Halás[3]

***Abstract*— In this paper, we investigate the practical problems of design and digital implementation of fractional-order PID controllers used for fluid level control in a system of coupled tanks. We present a method for obtaining the PI$^\lambda$D$^\mu$ controller parameters and describe the steps necessary to obtain a digital implementation of the controller. A real laboratory plant is used for the experiments, and a hardware realization of the controller fit for use in embedded applications is proposed and studied. The majority of tasks is carried out by means of the FOMCON ("Fractional-order Modeling and Control") toolbox running in the MATLAB computing environment.**

I. Introduction

Today, fractional-order calculus is an actively researched topic. The number of applications where it is used rapidly grows [1], [2]. The reason for this is the possibility to obtain accurate mathematical models of real objects, especially if those objects exhibit memory, hereditary, or self-similarity properties. Our interest in fractional-order calculus lies in the field of control system design. With enhanced modeling opportunities it is possible to develop novel robust control techniques. One notable example is CRONE control [3], [4]. It is also possible to harness the power of fractional operators to update the classical controllers, such as the PID controller and lead-lag compensator. The fractional-order counterparts of these controllers, namely the PI$^\lambda$D$^\mu$ controller and fractional lead-lag compensator, were presented in [5]–[7]. These fractional-order controllers allow to significantly improve control loop performance. Some efficient design techniques have been proposed over the years and are summarized in [2], [8].

However, the benefits of using accurate models and robust controllers come at the expense of complicated computations. The MATLAB software suite is a very good choice for working with complicated mathematical models and it is widely used in control engineering. There exist several toolboxes, specifically devoted to fractional calculus based control [3], [9].

In this work, we will use the FOMCON toolbox [10], [11] to design fractional PID controllers for a fluid tank system. This system has deep roots in the industry. It is also a valuable laboratory tool for teaching control engineering. Additionally, we will explore the possibilities of using the obtained fractional PID controllers in embedded applications.

The paper is organized as follows. In Section II a brief introduction to fractional control is provided. In Section III we describe the plant and corresponding model used in our controller design experiments. In Section IV a fractional PID controller design method is outlined. Then, in Section V we provide the description of the fractional PID digital implementation and present the experimental platform. Experimental results follow in Section VI. Section VII is devoted to discussion. Finally, in Section VIII conclusions are outlined.

II. Brief Introduction to Fractional Control

Fractional calculus is a generalization of differential and integral operators to a non-integer order fundamental operator ${}_a\mathscr{D}_t^\alpha$, where α is the operator order and a, t denote the limits of the operation. The continuous integro-differential operator of order $\alpha \in \mathbb{R}$ is defined in the following way

$${}_a\mathscr{D}_t^\alpha = \begin{cases} \mathrm{d}^\alpha/\mathrm{d}t^\alpha & \alpha > 0, \\ 1 & \alpha = 0, \\ \int_a^t (\mathrm{d}\tau)^{-\alpha} & \alpha < 0. \end{cases} \tag{1}$$

Several definitions of the fractional integro-differential operator exist. We, however, restrict our attention to one particular definition, since it is used throughout this work for evaluating fractional-order systems. This is the Grünwald-Letnikov definition [2] and it is as follows

$${}_a\mathscr{D}_t^\alpha f(t) = \lim_{h\to 0} \frac{1}{h^\alpha} \sum_{j=0}^{k} (-1)^j \binom{\alpha}{j} f(t-jh), \tag{2}$$

where $a = 0$, $t = kh$, k is the number of steps, and h is the step size. Assuming zero initial conditions, the Laplace transform of the fractional derivative with $\alpha \in \mathbb{R}^+$ is given by

$$\int_0^\infty \mathrm{e}^{-st}\, {}_0\mathscr{D}_t^\alpha f(t)\mathrm{d}t = s^\alpha F(s), \tag{3}$$

where $s = j\omega$ is the Laplace transform variable. Thus a fractional-order differential equation can be expressed in transfer function form in the Laplace domain as follows

$$G(s) = \frac{b_m s^{\beta_m} + b_{m-1} s^{\beta_{m-1}} + \cdots + b_0 s^{\beta_0}}{a_n s^{\alpha_n} + a_{n-1} s^{\alpha_{n-1}} + \cdots + a_0 s^{\alpha_0}}. \tag{4}$$

The control law of the PI$^\lambda$D$^\mu$ controller, where the integral component is of order λ and the derivative component is of

[1]Aleksei Tepljakov and Eduard Petlenkov are with the Department of Computer Control, Tallinn University of Technology, Tallinn, 19086, Estonia, {aleksei.tepljakov, eduard.petlenkov} at dcc.ttu.ee.

[2]Juri Belikov is with the Institute of Cybernetics, Tallinn University of Technology, Akadeemia tee 21, 12618, Tallinn, Estonia, jbelikov at cc.ioc.ee.

[3]Miroslav Halás is with Institute of Control and Industrial Informatics, Slovak University of Technology, Ilkovičova 3, 81219 Bratislava, Slovakia miroslav.halas at stuba.sk.

order μ, has the following form in the time domain

$$u(t) = K_p \cdot e(t) + K_i \cdot \mathscr{I}_t^{\lambda} e(t) + K_d \cdot \mathscr{D}_t^{\mu} e(t), \quad (5)$$

where $e(t)$ is the error signal and the fractional-order integral is defined as $\mathscr{I}_t^{\alpha} x(t) = \mathscr{D}_t^{-\alpha} x(t)$. In the Laplace domain, the transfer function corresponding to the parallel form of the fractional-order PID controller is the following

$$C(s) = K_p + \frac{K_i}{s^{\lambda}} + K_d \cdot s^{\mu}. \quad (6)$$

It can be easily seen, that in the frequency domain this controller has obvious advantages over the classical one. By definition, $s^{\alpha} = (j\omega)^{\alpha}$ and thus more freedom in changing the shape of the response is achieved. This is a very important controller design method that is frequently used.

Finally, it should be mentioned that in practice fractional-order operator approximations are often used. This allows to overcome the problem of the infinite memory requirement that the fractional-order systems possess. A very good approximation technique, whereby the fractional derivatives are approximated by integer-order ones, is the Oustaloup recursive filter method [4] and its modification [2], [12]. Once a continuous-time approximation is obtained, it is possible to compute the corresponding discrete-time model using a suitable method. A good summary of these methods is provided in [1], [13].

III. Description of the Coupled Tank System

The laboratory plant used in this work is a coupled fluid tank system shown in Fig. 1. The plant is conveniently connected to the PC via USB for data transfer. A Simulink library for real-time control of the plant is available.

Fig. 1. The laboratory plant

In the continuous-time domain the plant model can be described by the following differential equations

$$\begin{aligned} \dot{x}_1 &= \frac{1}{A} u_1 - d_{12} - w_1 c_1 \sqrt{x_1}, \quad (7) \\ \dot{x}_2 &= \frac{1}{A} u_2 + d_{12} - w_2 c_2 \sqrt{x_2}, \end{aligned}$$

where x_1 and x_2 are levels of fluid in the tanks, A is the cross section of both tanks, c_1, c_2, and c_{12} are flow coefficients, u_1 and u_2 are pump powers, and

$$d_{12} = w_{12} \cdot c_{12} \cdot \mathrm{sign}(x_1 - x_2) \sqrt{|x_1 - x_2|}.$$

The coupled tank system has three valves. Each of them can be either fully opened or fully closed. This is governed in the model by factors w_1, w_2, and w_{12} with $w \in \{0, 1\}$.

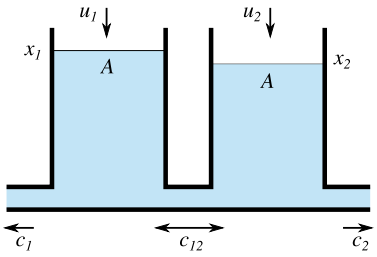


Fig. 2. Coupled tanks model

We now provide the discrete-time model of this system. Following the notation in [14] for $x(t+T)$ we shall write x^+, where T is the sampling period. The model is as follows

$$\begin{aligned} x_1^+ &= x_1 + \frac{T}{A} u_1 - d_{12} - w_1 c_1 T \sqrt{x_1}, \quad (8) \\ x_2^+ &= x_2 + \frac{T}{A} u_2 + d_{12} - w_2 c_2 T \sqrt{x_2}, \end{aligned}$$

where

$$d_{12} = T \cdot w_{12} \cdot c_{12} \cdot \mathrm{sign}(x_1 - x_2) \sqrt{|x_1 - x_2|}.$$

The flow coefficients for this particular plant were identified experimentally and are as follows: $c_1 = 0.0292$, $c_2 = 0.0259$, and $c_{12} = 0.0267$. The cross-section A of both tanks is $10.18 \cdot 10^{-4} \mathrm{m}^2$ and maximal pump power u_h for both pumps is $5.9174 \cdot 10^{-5} \mathrm{m}^3/\mathrm{s}$. The height of both tanks is $0.29\,\mathrm{m}$.

IV. Design of Fractional-order PID Controllers

Further, an outline of the fractional PID controller design method is provided. It is assumed that the plant can be modeled by a low order process model. Then, the process of obtaining the fractional PID controller parameters, i.e. component gains and integral/derivative orders, can comprise the following steps:

1) Approximate or identify the process by a simple process model, e.g. by a first-order plus dead time (FOPDT) model and obtain the initial PID parameters K_p, K_i, and K_d by using an established tuning formula, e.g. by using the Ziegler-Nichols method [15];
2) Use the obtained controller component gains as starting points for controller parameter optimization, based on particular design specifications and control loop performance criteria;
3) Evaluate the result and apply corrections to the controller parameters, if necessary.

In the FOMCON toolbox for MATLAB several tools exist to fulfill this task. It is possible to obtain a fractional-order model in form (4) from time-domain identification data using the time-domain identification tool `fotfid`, and it is also

possible to approximate fractional-order systems by process models and compute controller gains with the `iopid_tune` tool.

Once the initial parameters are obtained, it is possible to use the optimization tool `fpid_optim` to improve the robustness of the control loop by introducing the fractional orders λ and μ of the PI$^\lambda$D$^\mu$ controller and using numerical optimization. The cost function is computed by using one of the four performance indices (integral square error, integral absolute error, integral time square error, and integral time absolute error) from the simulated control loop response. Design specifications are set in the frequency domain [2]. A detailed description of specifications used in the tool is provided in [16] with relevant comments. The graphical user interface of the fractional-order PID controller optimization tool is shown in Fig. 3.

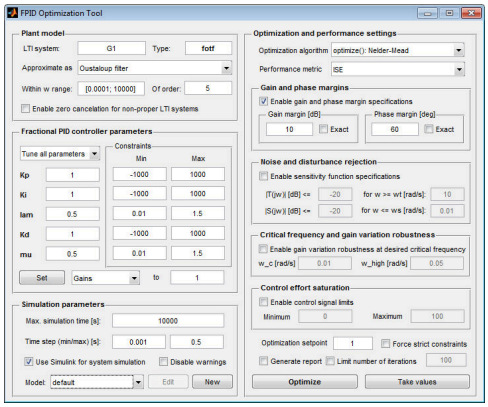


Fig. 3. Graphical user interface for fractional PID optimization tool

The tool allows to use Simulink for system simulation which allows to work with nonlinear systems. In this case the linear, time invariant system could still be used for computing the frequency-domain parameters of the open loop in order to assess the control system performance.

V. Controller Implementation and Experimental Platform

In this work we consider fractional-order system approximation to obtain the digital implementation of fractional-order PID controllers. Some previous implementation experiments are described in [17], [18]. It is important to take into account the limitations of target hardware, because fractional-order systems approximations are usually of high order. In this case we use the Atmel AVR 8-bit microcontroller ATmega8A for implementing the PI$^\lambda$D$^\mu$ controller. Several performance constraints should be considered:

- There is no specialized floating-point unit on this device, apart from a hardware multiplier, all calculations are done in software;
- Only single precision floating-point number format is available.

Therefore, the effects stemming from limited resolution floating-point computations will impair the controller realization. In [1], [18] it is suggested to implement the fractional controller by an IIR filter in canonical form

$$H(z) = \left(\sum_{j=0}^{N} b_j \cdot z^{-j} \right) \left(1 + \sum_{i=1}^{M} a_i \cdot z^{-i} \right)^{-1}, \quad (9)$$

where $N = M$ is the order of the filter.

However, in case of hardware with limited capabilities this may not be the best solution, since the stability of computation will be significantly reduced. One possibility is to use an IIR filter comprised of second-order sections [19], which has the following form:

$$H(z) = b_0 \prod_{k=1}^{N} \frac{1 + b_{1k} z^{-1} + b_{2k} z^{-2}}{1 + a_{1k} z^{-1} + a_{2k} z^{-2}}. \quad (10)$$

Such a realization is less sensitive to parameter variation due to coefficient quantization [20]. In canonical form this filter will require two memory elements per section for the delay terms which works favorably in case when memory is limited.

Let us now summarize the suggested implementation procedure:

1) Obtain a continuous-time approximation of the fractional-order PID controller by using the Oustaloup recursive filter method in a suitable frequency range, observing the resulting controller frequency-domain characteristics and comparing them to the ideal response;
2) Use a discretization method to obtain the filter discrete representation in the z-domain, check the frequency-domain characteristics;
3) Factor the coefficients of the resulting filter to obtain the second-order sections form, in case of poor coefficient scaling, try different approximation parameters in step 1.

In FOMCON, it is possible to use an experimental tool `impid` for this task. The MATLAB command `zp2sos()` can convert the zero-pole-gain representation, which is preferred due to enhanced accuracy, into second-order section form. For discretization purposes the `c2d()` command can be used with `'matched'` method [21].

Once the implementation is obtained, it should be transferred to the microcontroller. In this work the filter is hard-coded into the microcontroller firmware.

Next, let us describe the prototyping platform used in the controller implementation experiments [22]. The schematic diagram of the control loop is given in Fig. 4. The data acquisition board is connected to the computer via RS232 and in this configuration is capable of 5kSPS data rates with 8 bit sample resolution. On the computer, real-time control loop simulation is done by means of the Real-Time Windows Target toolbox for MATLAB. The resolution of analog-to-digital and digital-to-analog converters of the controller prototype is fixed at 8 bit.

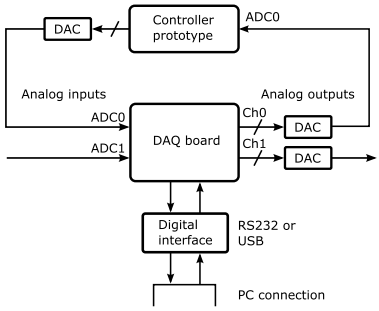


Fig. 4. Hardware loop connection diagram used in controller implementation experiments

VI. Experimental Results

Three types of experiments are considered:

1) Simulation of the fluid tank system model in MATLAB/Simulink used to optimize the controller parameters;
2) Control of the real plant from Simulink, the plant is connected to the computer via USB, no additional equipment is used;
3) Control of the model in (8) running in Simulink in the configuration depicted in Fig. 4.

Note, that the validation of the controller prototype is only done by using the plant model. The closed-loop behavior of the digital fractional PID controller prototype and the laboratory plant is yet to be investigated.

A. Single Tank

In this experiment, our goal is to control the level in a single tank, that is the valve connecting the two tanks is closed, i.e. in models (7) and (8) we have $w_{12} = 0$. Next, we describe the procedure of tuning the PI$^\lambda$D$^\mu$ controller to control the real plant.

First, a fractional-order LTI model is identified from the step experiment. The identification data is recorded from the real plant and is transformed such that both input and output signals are in range $u, y \in [0, 1]$. The obtained model is as follows

$$G_1(s) = \frac{1}{5.3538s^{1.0109} + 0.4126} \mathrm{e}^{-1s}. \qquad (11)$$

It can be seen, that this model tends to an integer-order FOPDT model. The latter can be approximated using `iopid_tune` tool in FOMCON as

$$\hat{G}_1(s) = \frac{2.43567}{1 + 12.5914s} \mathrm{e}^{-1.0779s} \qquad (12)$$

and integer-order PID controller gains can be obtained using the Ziegler-Nichols method as $K_p = 5.75515$, $K_i = 2.66962$, $K_d = 3.10175$.

The next step is to use the fractional-order PID controller optimization tool using the obtained integer-order gains as initial values. The model (4) is approximated by a refined Oustaloup filter of order $N = 5$ within $\omega = [0.0001, 10000]\,\mathrm{rad/s}$. The chosen performance metric is integral absolute error (IAE). Simulink is used for system simulation and control effort saturation is considered. Gain and phase margins are specified as $G_m = 15\,\mathrm{dB}$ and $\varphi_m = 90^\circ$, respectively. Optimization set point is 0.5. Finally, the gains and orders λ, μ are limited to the following ranges: $K_p = [0.1, 10], K_i = [0.001, 10], \lambda = 1, K_d = [0.1, 10], \mu = [0.5, 0.9]$. The selection of ranges is derived from considering the behavior of the fractional-order control actions in the time domain. After a 100 iterations the following results are obtained:

$$K_p = 1.1434, \quad K_i = 0.079556, \quad K_d = 0.89839, \quad \lambda = 1, \quad \mu = 0.80468.$$

Upon implementing this discrete controller in MATLAB with sampling time $T_s = 0.1\mathrm{s}$ and testing the control loop in real time with the laboratory plant, these parameters resulted in a significant overshoot of the controlled level. After manually shifting the points $K_p' = 2K_p$ and $K_d' = 2K_d$ and further optimization, the following PID$^\mu$ controller was designed:

$$K_p = 2.6282, \quad K_i = 0.097128, \quad K_d = 1.26, \quad \lambda = 1, \quad \mu = 0.78894. \qquad (13)$$

In Fig. 6 the response of the control system in presence of random measurement errors caused by physical interference can be observed. It can be seen, that the system easily recovers from such disturbance. The set point is $y = 0.2\mathrm{m}$.

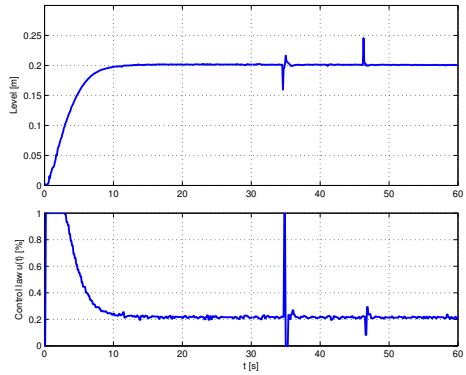


Fig. 5. Level control with measurement noise

Next, we consider reference tracking. In this case, we also test the controller prototype using the obtained parameters. The set point changes from $0.2\mathrm{m}$ to $0.15\mathrm{m}$ at the 40th second of the simulation. The results obtained are given in Fig. 6.

It can be seen, that the response of the control loop with the real plant is different from those, where a model was used. Therefore, the model should be revised accordingly.

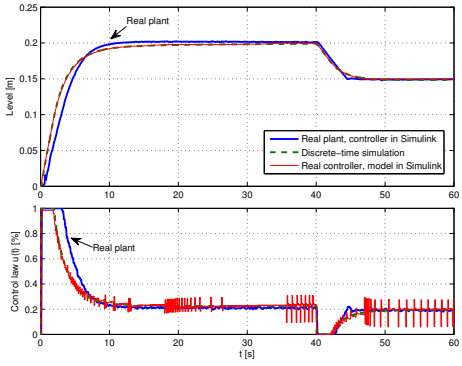


Fig. 6. Level control with set point change

The controller prototype implemented using the method described in Section V performs as expected. The corresponding control system response closely matches the simulated one. However, the control law produced by the physical controller suffers from limit cycles. This is due to the relatively low sample resolution which results in quantization [19], [20]. It is possible to eliminate the oscillations by increasing the sample resolution.

B. Coupled Tanks

This time the task is to control a level of the first tank in a system of coupled tanks, when in (7) and (8) we have $w_1 = w_{12} = 1$, and the last valve w_2, which can open at any time, is seen as disturbance. In [14] a nonlinear control design approach was used, in this work we use a linear PI$^\lambda$D$^\mu$ controller.

We identify the real plant from a step experiment with $w_1 = w_{12} = 1, w_2 = 0$. The resulting fractional-order model is described by a transfer function

$$G_2 = \frac{1}{7.3986s^{0.9455} + 0.4095} \mathrm{e}^{-0.1s}. \qquad (14)$$

We notice, that this model no longer tends to exhibit integer-order dynamics as in case of (11).

Due to the value of the delay term the basic tuning formulae for integer-order PID tuning do not provide feasible results. However, it is possible to select some starting point manually and run optimization several times. However, it is important to choose the correct frequency domain specifications to ensure control system stability. In our case the goal is to minimize the impact of disturbance, so constraints on the sensitivity functions could be imposed. Our choice is such that $|T(j\omega)| \leq -20\,dB, \forall\omega \geq 10\,\mathrm{rad/s}$ and $|S(j\omega)| \leq -20\,dB, \forall\omega \leq 0.1\,\mathrm{rad/s}$. The gain and phase margins are set to $G_m = 10\,\mathrm{dB}$ and $\varphi = 60°$, respectively. Additionally, due to the same problem as in the previous experiment, in order to limit the overshoot, the upper bound of control signal saturation was lowered from 100% to 60%. Using the IAE performance metric we finally arrive at the following PI$^\lambda$D$^\mu$ controller parameters by optimizing the response of the nonlinear system in Simulink:

$$K_p = 6.9514, \quad K_i = 0.13522, \quad K_d = -0.99874, \quad \lambda = 0.93187, \quad \mu = 0.29915. \qquad (15)$$

Next, the discrete approximation is obtained. Note, that in case of this controller it is very important to choose the proper frequency range of approximation. The suitable range for a refined Oustaloup filter of order 5 was found such that $\omega = [0.0001, 100]\,\mathrm{rad/s}$. The sampling time is $T_s = 0.2\mathrm{s}$. The responses of the three control systems in different configurations are depicted in Fig. 7. The set point is once again fixed at $y = 0.2\mathrm{m}$. Additional disturbance is such that the second pump is switched on at full capacity at $t = 76.4\,\mathrm{s}$ and is turned off at $t = 80.6\,\mathrm{s}$.

Once again we observe that the behavior of the real plant differs from the simulated one while the responses of the discrete controller in Simulink and the digital controller are virtually identical. The control law oscillations of the digital controller are of smaller amplitude than in the previous experiment with the single tank.

The obtained control system is capable of maintaining the set level with a tolerance of 5% in the presence of disturbances discussed above.

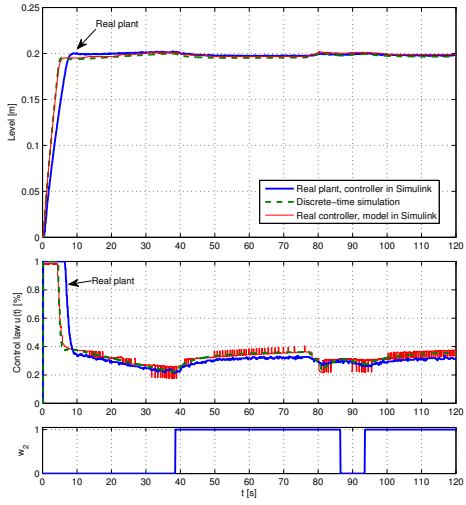


Fig. 7. Level control in a system of coupled tanks

Finally, we present some comments related to the computational capacity of the Atmel AVR ATmega8A based controller prototype. Table I contains information about the time required to compute a single output sample. It can be seen, for example, that using an IIR filter with 7 sections allows the sample time to be as low as 600 μs.

TABLE I

ATMEGA8A OUTPUT SAMPLE COMPUTATION

No. of IIR second-order sections	Clock cycles required per sample computation	Sample computation time at 16 MHz
7	9057	566.07 μs
14	18208	1.138 ms

VII. DISCUSSION

During our work we have run into several problems briefly outlined next.

- The difference in the model and the real plant is quite evident and should be eliminated. A more thorough parameter identification procedure may be needed. Once a more accurate model is obtained, the results of controller design via simulation based optimization should also improve.
- The limited sample resolution of the controller prototype introduces limit cycles of a substantial amplitude into the control signal due to quantization. The results of further simulations of these effects suggest that increasing the sample resolution to 10 bit significantly reduces the oscillations.
- Difficulties may arise in discrete-time approximation of a fractional-order controller. Careful choice of approximation parameters may resolve the issue.

While optimization based tuning can provide feasible results, efforts should be devoted into developing analytical tuning rules for the fractional-order PID controller. Implementation of efficient auto-tuning is also highly desired.

VIII. CONCLUSIONS

In this paper we presented a method for fractional-order PID controller design in the MATLAB/Simulink environment using the FOMCON toolbox. We have also provided a description of a procedure that can be used to obtain digital implementations of fractional-order controllers and discussed the associated issues. Illustrative examples were provided to support these methods. The proposed tuning method is quite general and is thus suitable for a wide variety of plants. In this work, we have obtained results comparable to those achieved by nonlinear control designed for the coupled tank system in [14], [23]. Further work will be devoted to improving the obtained results.

ACKNOWLEDGMENTS

This work was partially supported by the Governmental funding project no. SF0140113As08, the Estonian Research Council Grant no. 8738, and the Estonian Doctoral School in Information and Communication Technology through the interdisciplinary project FOMCON.

We would like to thank Associate Professor Miroslav Halas from Slovak University of Technology for the opportunity to hold a number of experiments with the coupled tanks laboratory plant.

REFERENCES

[1] Y. Q. Chen, I. Petráš, and D. Xue, "Fractional order control - a tutorial," in *Proc. ACC '09. American Control Conference*, 2009, pp. 1397–1411.

[2] C. A. Monje, Y. Chen, B. Vinagre, D. Xue, and V. Feliu, *Fractional-order Systems and Controls: Fundamentals and Applications*, ser. Advances in Industrial Control. Springer Verlag, 2010.

[3] A. Oustaloup, P. Melchior, P. Lanusse, O. Cois, and F. Dancla, "The CRONE toolbox for Matlab," in *Proc. IEEE Int. Symp. Computer-Aided Control System Design CACSD 2000*, 2000, pp. 190–195.

[4] A. Oustaloup, F. Levron, B. Mathieu, and F. M. Nanot, "Frequency-band complex noninteger differentiator: characterization and synthesis," *IEEE Trans. Circuits Syst. I*, vol. 47, no. 1, pp. 25–39, 2000.

[5] I. Podlubny, L. Dorčák, and I. Kostial, "On fractional derivatives, fractional-order dynamic systems and $PI^{\lambda}D^{\mu}$-controllers," in *Proc. 36th IEEE Conf. Decision and Control*, vol. 5, 1997, pp. 4985–4990.

[6] I. Podlubny, "Fractional-order systems and $PI^{\lambda}D^{\mu}$-controllers," *IEEE Trans. Autom. Control*, vol. 44, no. 1, pp. 208–214, 1999.

[7] D. Xue and Y. Chen, "A comparative introduction of four fractional order controllers," in *Proceedings of the 4th World Congress on Intelligent Control and Automation*, 2002, pp. 3228–3235.

[8] C. Monje, B. Vinagre, V. Feliu, and Y. Chen, "Tuning and auto-tuning of fractional order controllers for industry applications," *Control Engineering Practice*, vol. 16, no. 7, pp. 798–812, 2008.

[9] D. Valério. (2005) Toolbox ninteger for MatLab, v. 2.3. [Online]. Available: http://web.ist.utl.pt/duarte.valerio/ninteger/ninteger.htm

[10] A. Tepljakov, E. Petlenkov, and J. Belikov, "FOMCON: a MATLAB toolbox for fractional-order system identification and control," *International Journal of Microelectronics and Computer Science*, vol. 2, no. 2, pp. 51–62, 2011.

[11] ——. (2011) FOMCON toolbox. [Online]. Available: http://www.fomcon.net/

[12] D. Xue, Y. Chen, and D. P. Atherton, *Linear Feedback Control: Analysis and Design with MATLAB (Advances in Design and Control)*, 1st ed. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2008.

[13] B. M. Vinagre, I. Podlubny, A. Hernández, and V. Feliu, "Some approximations of fractional order operators used in control theory and applications," *Fractional Calculus & Applied Analysis*, vol. 3, pp. 945–950, 2000.

[14] M. Halás, "Discrete-time solution to the disturbance decoupling problem of coupled tanks," in *Proc. of the 18th International Conference on Process Control*, M. Fikar and M. Kvasnica, Eds., Tatranská Lomnica, Slovakia, 2011, pp. 162–167.

[15] K. Åström and T. Hägglund, *PID Controllers: Theory, Design, and Tuning*, 2nd ed. The Instrumentation, Systems, and Automation Society (ISA), 1995.

[16] A. Tepljakov, E. Petlenkov, and J. Belikov, "A flexible MATLAB tool for optimal fractional-order PID controller design subject to specifications," in *Proceedings of the 31st Chinese Control Conference*, W. Li and Q. Zhao, Eds., Hefei, Anhui, China, 2012, pp. 4698–4703.

[17] I. Petras, S. Grega, and L. Dorcak, "Digital fractional order controllers realized by PIC microprocessor: Experimental results," in *Proc. of the ICCC'2003 conference*, High Tatras, Slovak Republic, 2003, pp. 873–876.

[18] I. Petráš, "Practical aspects of tuning and implementation of fractional-order controllers," *ASME Conference Proceedings*, pp. 75–84, 2011.

[19] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Discrete-time signal processing (2nd ed.)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1999.

[20] S. Kuo and B. Lee, *Real-time digital signal processing: implementations, applications, and experiments with the TMS320C55X*. Wiley, 2001.

[21] G. Franklin, J. Powell, and M. Workman, *Digital Control of Dynamic Systems*. Ellis-Kagle Press, 2006.

[22] A. Tepljakov, E. Petlenkov, and J. Belikov, "Implementation and real-time simulation of a fractional-order controller using a MATLAB based prototyping platform," in *Proc. 13th Biennial Baltic Electronics Conference*, 2012, pp. 145–148.

[23] V. Žilka, M. Halás, and M. Huba, "Nonlinear controllers for a fluid tank system," in *Computer Aided Systems Theory - EUROCAST 2009*, R. Moreno-Díaz, F. Pichler, and A. Quesada-Arencibia, Eds. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 618–625.

Publication 3

Reference

A. Tepljakov, E. Petlenkov, and J. Belikov, "Efficient analog implementations of fractional-order controllers," in Proc. of the 14th International Carpathian Control Conference (ICCC), 2013, pp. 377–382.

Abstract

Fractional-order calculus is a very useful tool which extends classical, integer-order calculus and is used in contemporary modeling and control applications. It allows to describe dynamical systems more accurately, as well as gain valuable insight into some specific, memory, hereditary, and self-similarity properties of such systems. Fractional-order controllers, e.g. the $PI^{\lambda}D^{\mu}$ controller and fractional lead-lag compensator, based on the added flexibility of fractional-order operators, are capable of superior performance compared to their integer-order counterparts. However, there exist multiple issues associated with the implementation of these fractional-order systems. In this work we consider the problem of efficient analog realization of fractional-order controllers. We investigate the possibilities of network generation from fractional-order controller approximations derived using different methods proposed over the years. We consider the problem of practical feasibility of the resulting network as well as the preservation of controller performance specifications. Suitable tools, developed for the MATLAB environment in the context of the FOMCON ("Fractional-order Modeling and Control") toolbox, are presented and discussed. Results of relevant experiments, encompassing the simulation of the designed circuit are provided.

Efficient Analog Implementations of Fractional-Order Controllers

Aleksei Tepljakov and Eduard Petlenkov
Department of Computer Control
Tallinn University of Technology
Ehitajate tee 5, 19086 Tallinn, Estonia
e-mail: aleksei.tepljakov@dcc.ttu.ee
and eduard.petlenkov@dcc.ttu.ee

Juri Belikov
Institute of Cybernetics
Tallinn University of Technology
Akadeemia tee 21, 12618 Tallinn, Estonia
e-mail: jbelikov@cc.ioc.ee

***Abstract*—Fractional-order calculus is a very useful tool which extends classical, integer-order calculus and is used in contemporary modeling and control applications. It allows to describe dynamical systems more accurately, as well as gain valuable insight into some specific, memory, hereditary, and self-similarity properties of such systems. Fractional-order controllers, e.g. the PI$^\lambda$D$^\mu$ controller and fractional lead-lag compensator, based on the added flexibility of fractional-order operators, are capable of superior performance compared to their integer-order counterparts. However, there exist multiple issues associated with the implementation of these fractional-order systems. In this work we consider the problem of efficient analog realization of fractional-order controllers. We investigate the possibilities of network generation from fractional-order controller approximations derived using different methods proposed over the years. We consider the problem of practical feasibility of the resulting network as well as the preservation of controller performance specifications. Suitable tools, developed for the MATLAB environment in the context of the FOMCON ("Fractional-order Modeling and Control") toolbox, are presented and discussed. Results of relevant experiments, encompassing the simulation of the designed circuit are provided.**

***Index Terms*—fractional-order calculus, fractional control, approximation methods, electrical network, circuit synthesis**

I. Introduction

Fractional-order calculus is a mathematical concept and formalism, which allows to accurately model real-life phenomena, especially such that exhibit memory and hereditary properties [1], [2]. Naturally, fractional-order calculus has found its way into contemporary control applications. Conventional industrial controller models, such as the PID controller and lead-lag compensator, were generalized to benefit from the non-integer modeling approach [3], [4], [5], [6].

Due to the inherent complexity of fractional-order computational models, efficient analog implementation of fractional-order systems and controllers may be difficult. While specific discrete electronic components are being actively developed and tested [7], [8], electrical network based approximations of fractional-order elements, called *fractances* [9], can be effectively used. Apart from control applications, fractance elements can also be used in simulations of complex processes.

The foundations for some of the modern analog realizations of fractance circuits were laid in [10], [11], [12]. One of the most important developments in this field is perhaps the work of Alain Oustaloup and the CRONE research group [13], [1]. More recently the problem of analog realization of fractance elements has been investigated in [14], [15], [16], [17], [18].

In this paper, we consider the problem of efficient network generation for fractance circuits derived from earlier works and based on frequency domain analysis. Our main motivation is to develop a systematic approach to facilitate research and development of fractional-order signal processing and control applications involving fractance component approximations. The required efficiency is achieved by integrating this knowledge into a fully-featured software solution thereby allowing to quickly obtain fractance circuit realizations. We focus on the realizability of the resulting network and the preservation of control loop performance specifications. The results of this work can be applied in manufacturing of fractance circuits, including discrete component based active filter realizations. The main contribution is a MATLAB toolset designed for working with fractance networks in application to analog realization of fractional-order system models and controls.

The paper is organized as follows. In Section II a brief introduction to fractional-order control theory is given. In Section III we present an overview of fractance circuit network synthesis for fractional-order controller implementation. We cover several approximation schemes and network structures. In Section IV we present our approach to systematization of currently available methods and describe the particular realization in MATLAB software in the context of the FOMCON toolbox [19], [20]. Illustrative examples follow in Section V. Some topics for discussion are provided in Section VI. Finally, conclusions are drawn in the last section.

II. Brief Introduction to Fractional Control

Fractional calculus is a generalization of differential and integral operators to a non-integer order operator ${}_a\mathscr{D}_t^\alpha$, where α is the order of the operator, a and t denote the limits of the operation. The operator is defined in the following way

$$ {}_a\mathscr{D}_t^\alpha = \begin{cases} \mathrm{d}^\alpha/\mathrm{d}t^\alpha & \alpha > 0, \\ 1 & \alpha = 0, \\ \int_a^t (\mathrm{d}\tau)^{-\alpha} & \alpha < 0. \end{cases} \tag{1} $$

978-1-4673-4490-6/13/$31.00 ©2013 IEEE

Assuming zero initial conditions, the Laplace transform of the fractional derivative with $\alpha \in \mathbb{R}^+$ is given by

$$\int_0^\infty \mathrm{e}^{-st}\, {}_0\mathscr{D}_t^\alpha f(t)\mathrm{d}t = s^\alpha F(s), \tag{2}$$

where $s = j\omega$ is the Laplace transform variable. Thus a fractional-order differential equation can be expressed in transfer function form in the Laplace domain as follows

$$G(s) = \frac{b_m s^{\beta_m} + b_{m-1} s^{\beta_{m-1}} + \cdots + b_0 s^{\beta_0}}{a_n s^{\alpha_n} + a_{n-1} s^{\alpha_{n-1}} + \cdots + a_0 s^{\alpha_0}}. \tag{3}$$

The fractional PI$^\lambda$D$^\mu$ controller [3], where λ and μ denote the orders of the integral and differential components, respectively, is represented as follows:

$$C(s) = K_p + \frac{K_i}{s^\lambda} + K_d \cdot s^\mu. \tag{4}$$

It can be seen, that this controller has advantages over the classical one since there is more freedom in shaping the response. In the frequency domain by varying the order γ of a fractional-order integrator (differentiator) one can achieve a constant decrement (increment) in the slope of the magnitude curve that equals 20γ dB/dec and a constant delay in the phase plot $\pi\gamma/2$ rad, where the values depend on the sign of γ.

The transfer function, corresponding to the fractional lead-lag compensator of order α, has the following form, involving an implicit expression:

$$C_L(s) = K\left(\frac{1+bs}{1+as}\right)^\alpha. \tag{5}$$

When $\alpha > 0$ we have the fractional zero and pole frequencies $\omega_z = 1/b$, $\omega_h = 1/a$ and the transfer function in (5) corresponds to a fractional lead compensator. For $\alpha < 0$, a fractional lag compensator is obtained. This type of controller has received a considerable amount of attention and is useful in a number of control applications [21], [22], [23].

III. Electronic Realization of Fractional-order Controllers

In the following, we provide a summary of the network structures and corresponding synthesis methods used in this work with relevant comments.

Over the years, several methods involving use of Cauer and Foster canonical network forms were proposed [9]. Two of these structures will be used in this work. The Cauer I form RC circuit, presented in Fig. 1, has the following impedance, obtained by applying continued fraction expansion[24]:

$$Z_{RC}(s) = R_1 + \cfrac{1}{C_2 s + \cfrac{1}{R_3 + \cfrac{1}{C_4 s + \cdots}}}. \tag{6}$$

The Foster II form RC circuit is given in Fig. 2. The corresponding admittance can be expressed by means of partial fraction expansion in the following way:

$$Y_{RC}(s) = \frac{1}{Z_{RC}(s)} = \frac{1}{R_p} + C_p s + \sum_{i=1}^{n} \frac{K_i s}{s+\sigma_i}, \tag{7}$$

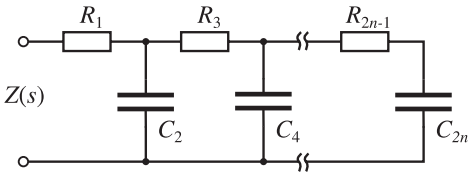


Fig. 1. Cauer I form canonical RC network

where $K_i = 1/R_i$ and $\sigma_i = K_i/C_i$.

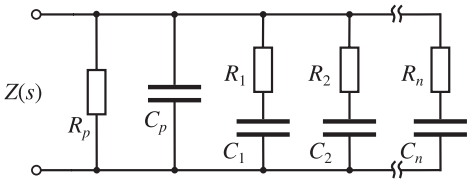


Fig. 2. Foster II form canonical RC network

In general, in order to obtain the fractance network component values there are several choices:

- Use a suitable approximation technique to obtain the impedance in form $Z(s)$ and develop it into a suitable expansion, thus obtaining the R, C, or L component values. The values of the components are not well-scaled and negative values may be obtained, in which case one would need to use negative impedance converters [9].
- Use constrained optimization to identify the network.
- Apply a method to derive the required component values directly in a much more controlled manner.

In [10], a RC driving-point immitance approach, applied to a Foster II canonical form RC network, was proposed to achieve a constant phase angle φ over a designated frequency range. Useful relations were highlighted to determine the values of the R and C subsequent components. A similar approach was used in [25] and later in [16], [17], [18]. In what follows, we briefly describe this method and provide its application to fractional-order system and controller implementation.

Given generation parameters α (fractional operator order), $\Delta\varphi$ (phase ripple), R_1 and C_1 (base resistor and capacitor values, which are chosen according to the frequency range of interest). According to these values, the following parameters are calculated:

$$\eta \approx \frac{0.24}{1+\Delta\varphi}, \quad a = 10^{\alpha\log(\eta)}, \quad b = \frac{\eta}{a}, \tag{8}$$

where $0 < a < 1$ and $0 < b < 1$. The values of m resistors and m capacitors, comprising the network, are then obtained by using the following set of synthesis formulae:

$$R_k = R_1 a^{k-1}, \quad C_k = C_1 b^{k-1}, \quad k = 1, 2, \ldots, m. \tag{9}$$

The values of R_p and C_p are obtained using

$$R_p = R_1 \frac{1-a}{a}, \quad C_p = C_1 \frac{b^m}{1-b}. \tag{10}$$

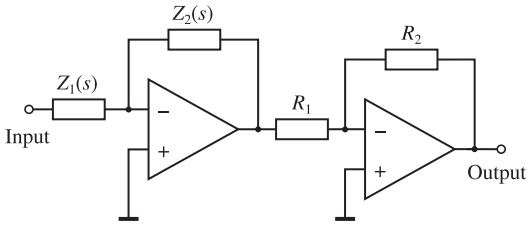


Fig. 3. Analog approximation of a fractional-order operator

Finally, we need to take into account the correction gain K. If the approximated operator of order α is given by an integer-order transfer function $G(\alpha, s)$ and the approximation yields an admittance $Y(j\omega)$ in (7), then at a frequency

$$\omega_m = \frac{\sqrt{a}}{R_1 C_1 \eta^{\lfloor m/2 \rfloor - 1}}, \tag{11}$$

where $\lfloor \cdot \rfloor$ denote the floor function, the gain is computed as

$$K = \frac{|G(\alpha, j\omega_m)|}{|Z(j\omega_m)|}, \tag{12}$$

where $Z(j\omega_m)$ is the impedance of the obtained approximation.

This method can be effectively applied to implement the fractional PID controller. We may use the synthesis procedure in (8)–(12) separately for the fractional integrator and fractional differentiator to arrive at two fractance networks, which will form the controller in a general active filter configuration as illustrated in Fig. 3. $Z_1(s)$ and $Z_2(s)$ should be chosen accordingly and reduced to a trivial resistance if need be.

We conclude this section by summarizing the steps required to implement a fractional lead-lag compensator. The same synthesis formulae can be used, however:

- The frequency range of approximation, determined by $\tau = R_1 C_1$, must be correctly chosen;
- The gain of the differential or integral component must be corrected by a factor $1/\omega_z^{\alpha}$ such that

$$G_c(\alpha, s) = \omega_z^{-\alpha} G(\alpha, s), \tag{13}$$

where ω_z is the zero frequency in (5), $G_c(\alpha, s)$ and $G(\alpha, s)$ are integer-order transfer functions, approximating the fractional lead-lag compensator and the corresponding fractional-order operator, respectively.

IV. Implementation in MATLAB

Our goal was to obtain a general enough approach to systematization of existing network topologies and their generation. In FOMCON, fractance network synthesis and analysis is handled by means of a central component—an object, containing complete circuit information. The object contains references to

- particular network structures (MATLAB functions), however complex, which return computed network transfer functions (impedance values),
- corresponding implementations (also MATLAB functions).

The idea is that a single structure can have several different implementations, including, e.g., optimization based ones. The relations are illustrated in Fig. 4.

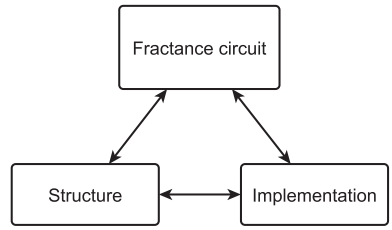


Fig. 4. Fractance, network structure, and implementation relations

This way one can implement a fractional PID controller and store it in a single object. For generality, we also consider the possibility of using inductive components in network structures.

Fractance is implemented by means of a class `frac_rcl()`, which has following properties:

- *model* — a model of the fractional-order system (`fotf` object);
- *structure* — network structure (Cauer, Foster, etc.);
- *implementation* — function that carries out the actual computation of the network component values;
- ω — frequency points used for model validation;
- *params* — parameters used for implementation and/or in the structure;
- K — network gain compensation factor(s);
- R, C, L — cell array with component value vectors, the size of the array is determined by the number of substructures within the main structure;
- *results* — implementation/validation results.

The first five parameters are used to create the `frac_rcl()` object. The following particular methods are implemented:

- `tf()`, `zpk()` — return the impedance $Z(s)$ in transfer function or zero-pole-gain format, corresponding to the fractance circuit, for network analysis. This can be used to automate, e.g. frequency response analysis.
- `prefnum()` — locates closest component values according to the preferred series and replaces network components accordingly. It is also possible to provide custom values of components so that the algorithm will choose the closest matches among a custom set of values. This can be necessary to analyze the changes of the system frequency characteristics due to variation of component values.
- `zscale()` — implementation of impedance scaling used to shift the values of discrete electronic components into the feasible domain.

Abstraction of fractance into a class has obvious benefits. The user has an option of writing implementations for existing

structures, or using the provided ones, while still using a single universal object to encompass the network. Some specific function naming schemes and coding conventions need to be followed by the programmer. Also, the programmer is responsible for documenting new, custom structures.

In the following, we illustrate the use of our approach.

V. Fractance Network Generation Examples

A. Fractional-order Low-pass Filter

In this example, we define a structure for a fractional low-pass filter

$$G(s) = \frac{1}{1 + 7s^{0.35}}$$

and provide the steps of the implementation procedure. Note, that this corresponds to the fundamental linear fractional-order differential equation and there are different means of implementation [14]. However, our goal here is to illustrate how structures can be broken down into substructures. We proceed as follows:

- Choose an initial value for the resistor R in $RCs^{0.35}$ as $15k\Omega$;
- Consider implementing the operator $s^{0.35}$ by means of the Oustaloup recursive filter;
- Use continued fraction expansion to arrive at a Cauer I type RC network;
- Use impedance scaling.

With this in mind, we build a custom MATLAB structure function `frac_struct_rc_lowpass_c1()`, which, in turn, uses a structure function `frac_struct_rc_cauer1()` and the corresponding implementation `frac_imp_rc_lowpass_oc1()`. Then, we use the following set of commands:

```
G1 = fotf('1','1+7s^0.35');

params       = struct;
params.R1    = 15e3;
params.N     = 5;

w = logspace(-5,2,200);

imp1 = frac_rcl(G1, ...
       'frac_struct_rc_lowpass_c1', ...
       'frac_imp_rc_lowpass_oc1', ...
       w, ...
       params);
```

An object is returned, where R and C properties are cell arrays, the first one contains the chosen resistor R, and the second one has the R and C values of the fractional capacitor approximation. The values need to be scaled, so the following command is issued:

```
imp1 = imp1.zscale({[], 0.001});
```

Finally, we obtain a passive circuit, corresponding to the fractional-order low-pass filter, depicted in Fig. 5. The component values are within a more or less feasible region, although the implementation of this system may be difficult and require careful considerations to be made. The comparison of the ideal frequency response of the fractional-order low-pass filter versus the network approximation is given in Fig. 6. The frequency response for the network was obtained by simulating the network in LTSpice IV and imported into MATLAB for plotting. As it can be seen, the approximation is valid within the approximate range $\omega = (10^{-3}; 10)$ rad/s.

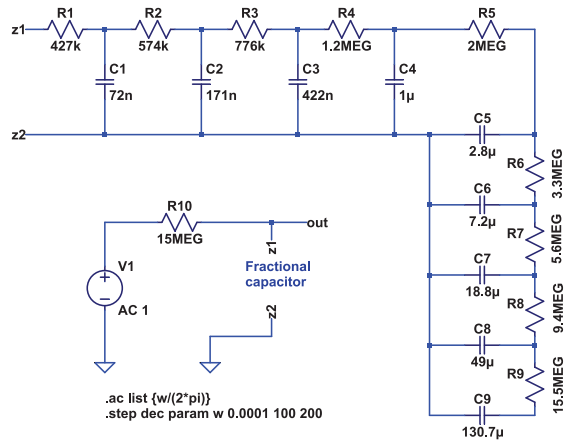


Fig. 5. Fractional low-pass filter schematic in LTSpice IV

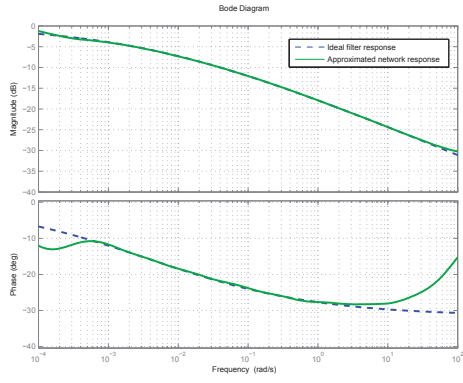


Fig. 6. Fractional low-pass filter frequency response: ideal vs. approximated

B. Fractional Lead-Lag Compensator

In this example, our goal is to obtain an analog implementation a fractional controller for a model of a position servo

$$G(s) = \frac{1.4}{s(0.7s + 1)} \mathrm{e}^{-0.05s} \tag{14}$$

identified in [23]. The design specifications are as follows: phase margin $\varphi = 80°$, gain crossover frequency $\omega_{cg} = 2.2$ rad/s. In the same reference paper, a controller design was proposed, based on robustness considerations derived from the

desired frequency domain characteristics of the plant, in the form of a fractional lead compensator:

$$C(s) = \left(\frac{2.0161s+1}{0.0015s+1}\right)^{0.7020}.$$

We implement this controller using the method from Section III. We choose $R_1 = 200k\Omega$ and $C_1 = 1\mu$F. The basic structure is the Foster II form RC network and the implementation is done by means of the described algorithm. To obtain the differentiator, we use the property $Z_d(s) = 1/Z_i(s)$, where $Z_d(s)$ and $Z_i(s)$ correspond to impedances of a differentiator and an integrator, respectively. This is done by setting the impedances in Fig. 3 such that $Z_1(s) = Z_i(s)$ and $Z_2(s) = R_k$, where R_k serves as a gain correction resistor. To obtain the network approximation, the following MATLAB commands are executed:

```
b  = 2.0161; wz = 1/b;
alpha = 0.702;
Gc = fotf('s')^alpha / wz^alpha;

params = struct; params.R1 = 200e3;
params.C1 = 1e-6; params.N = 4;
params.varphi = 0.01;

imp2 = frac_rcl(1/Gc, ...
       'frac_struct_rc_foster2', ...
       'frac_imp_rc_foster2_abgen', ...
       logspace(-2,3,1000), ...
       params);
```

Note, that the transfer function approximation, corresponding to the controller, has to be obtained from the resulting circuit object by using

```
C = 1/zpk(imp2);
```

Following the successful generation of the network, the `prefnum()` command is issued, which requests setting the resistor values to the preferred series with 5% tolerance, and the capacitor values substituted for closest components out of the 10%-series. This is done as follows:

```
imp2 = imp2.prefnum('5%','10%',[],'5%');
```

Finally, the user may wish to display the bill of materials by using the function `engnum()`. For example, for the list of resistors one can use the command:

```
[vals, str] = engnum(imp2.R);
```

The variable `str` will contain string constants with the values of the resistors in the network:

```
'360 k'   '200 k'   '75 k'   '27 k'   '9.1 k'
```

The same could be applied to the gain setting resistor R_k, which in our case has the preferred value of $390k\Omega$. The resulting network is depicted in Fig. 7. The schematic is simplified for simulation in LTSpice IV.

In Fig. 8 a comparison of frequency responses of the ideal lead compensator, the corresponding fractional-order differentiator and the resulting network approximation can be

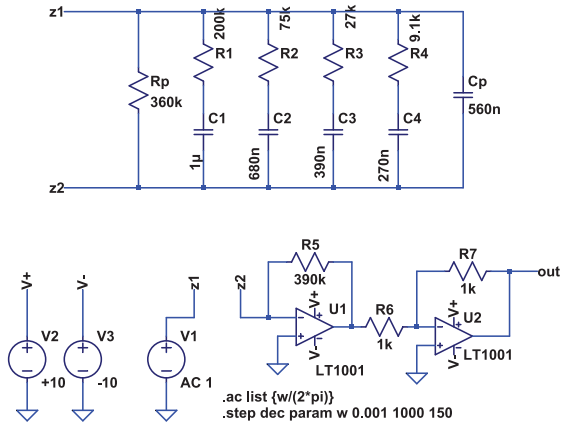


Fig. 7. Fractional lead-lag compensator realization schematic in LTSpice IV

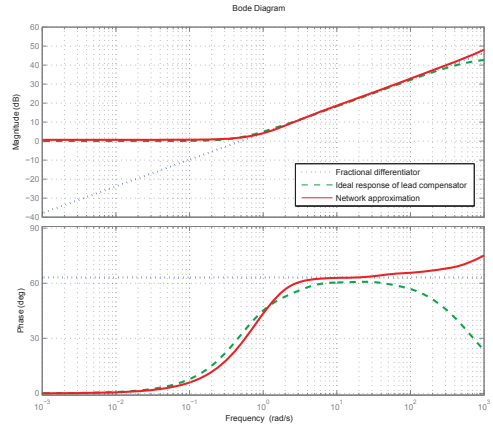


Fig. 8. Frequency response of fractional differentiator $0.6113 \cdot s^{0.702}$ vs. lead compensator vs. network approximation implementation

observed. As in the previous example, the frequency characteristics of the network were obtained from SPICE simulation and imported into MATLAB.

The open-loop frequency response of the control system is given in Fig. 9. A shift in the design specifications can be observed. Corrections may need to be made to particular component values or the overall network structure must be enhanced.

VI. Discussion

A necessary further step is to use the obtained network approximation to manufacture a hardware implementation of specific fractional-order systems and controllers. Then, means of improving the tool can be deduced from the practical perspective. Additionally, some issues discussed in this paper must also be addressed:

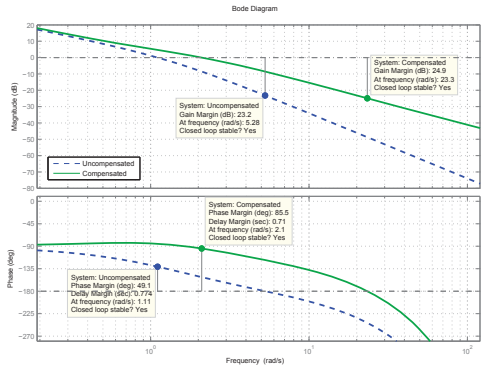


Fig. 9. Open-loop frequency response of the controlled system

- Selection of a network structure for a particular application;
- The tool must ensure the validity of the resulting network in the frequency range of interest.

One way to tackle the aforementioned problems is by means of carefully devised constrained optimization. This approach will be investigated in the future.

VII. Conclusions

In this paper, we have presented an approach to electrical network generation, suitable for implementing fractances in application to system modeling and control. A particular MATLAB realization, integrated into the FOMCON toolbox, was proposed. The presented solution offers means to generalize different network structures and implementations into a single object, which can be further manipulated to achieve a particular engineering goal. Further research in this area will take the following directions:

- Creation of a more diverse set of structures and implementations;
- Network structure enhancements for particular modeling and control tasks;
- Automation of identification of such an electrical network that corresponds to a set of given performance criteria;
- Implement automatic SPICE model generation;
- Creation of a more user friendly experience for using the proposed tool.

Acknowledgments

The work was supported by the European Union through the European Regional Development Fund, the target funding project SF0140113As08 of Estonian Ministry of Education and the Estonian Research Council grant N8738, and the Estonian Doctoral School in Information and Communication Technology through the interdisciplinary project FOMCON.

References

[1] A. Oustaloup, F. Levron, B. Mathieu, and F. M. Nanot, "Frequency-band complex noninteger differentiator: characterization and synthesis," *IEEE Transactions on Circuits and Systems—Part I: Fundamental Theory and Applications*, vol. 47, no. 1, pp. 25–39, 2000.

[2] Y. Q. Chen, I. Petráš, and D. Xue, "Fractional order control - a tutorial," in *Proc. ACC '09. American Control Conference*, 2009, pp. 1397–1411.

[3] I. Podlubny, L. Dorčák, and I. Kostial, "On fractional derivatives, fractional-order dynamic systems and PI$^{\lambda}$D$^{\mu}$-controllers," in *Proc. 36th IEEE Conf. Decision and Control*, vol. 5, 1997, pp. 4985–4990.

[4] I. Podlubny, "Fractional-order systems and PI$^{\lambda}$D$^{\mu}$-controllers," *IEEE Transactions on Automatic Control*, vol. 44, no. 1, pp. 208–214, 1999.

[5] ——, *Fractional differential equations*, ser. Mathematics in science and engineering. Academic Press, 1999.

[6] D. Xue and Y. Chen, "A comparative introduction of four fractional order controllers," in *Proceedings of the 4th World Congress on Intelligent Control and Automation*, 2002, pp. 3228–3235.

[7] I. Petráš, Y. Chen, and C. Coopmans, "Fractional-order memristive systems," in *Proc. IEEE Conf. Emerging Technologies & Factory Automation ETFA 2009*, 2009, pp. 1–8.

[8] D. Mondal and K. Biswas, "Performance study of fractional order integrator using single-component fractional order element," *IET Circuits, Devices & Systems*, vol. 5, no. 4, pp. 334–342, 2011.

[9] I. Podlubny, I. Petráš, B. M. Vinagre, P. O'Leary, and L. Dorčák, "Analogue realizations of fractional-order controllers," *Nonlinear Dynamics*, vol. 29, pp. 281–296, 2002.

[10] R. Morrison, "RC constant-argument driving-point admittances," *IRE Transactions on Circuit Theory*, vol. 6, no. 3, pp. 310–317, 1959.

[11] G. Carlson and C. Halijak, "Approximation of fractional capacitors $(1/s)^{1/n}$ by a regular Newton process," *IEEE Transactions on Circuit Theory*, vol. 11, no. 2, pp. 210–213, 1964.

[12] S. Roy, "On the realization of a constant-argument immittance or fractional operator," *Circuit Theory, IEEE Transactions on*, vol. 14, no. 3, pp. 264–274, september 1967.

[13] A. Oustaloup, J. Sabatier, and P. Lanusse, "From fractal robustness to the CRONE control," *Fractional Calculus and Applied Analysis*, vol. 2, no. 1, pp. 1–30, 1999.

[14] A. Charef, "Modeling and analog realization of the fundamental linear fractional order differential equation," *Nonlinear Dynamics*, vol. 46, pp. 195–210, 2006.

[15] ——, "Analogue realisation of fractional-order integrator, differentiator and fractional piλdμ controller," *IEE Proceedings-Control Theory and Applications*, vol. 153, no. 6, pp. 714–720, 2006.

[16] L. Dorčák, J. Terpák, I. Petráš, and F. Dorčáková, "Electronic realization of the fractional-order systems," *Acta Montanistica Slovaca*, vol. 12, no. 3, pp. 231–237, 2007.

[17] J. Valsa, P. Dvořá, and M. Friedl, "Network model of the CPE," *Radioengineering*, vol. 20, no. 3, pp. 619–626, September 2011.

[18] L. Dorčák, J. Valsa, J. Terpák, I. Petráš, and E. Gonzalez, "Comparison of the electronic realization of the fractional-order system and its model," in *Proc. 13th Int. Carpathian Control Conf. (ICCC)*, 2012, pp. 119–124.

[19] A. Tepljakov, E. Petlenkov, and J. Belikov, "FOMCON: a MATLAB toolbox for fractional-order system identification and control," *International Journal of Microelectronics and Computer Science*, vol. 2, no. 2, pp. 51–62, 2011.

[20] ——, "FOMCON: Fractional-order modeling and control toolbox for MATLAB," in *Proc. 18th Int Mixed Design of Integrated Circuits and Systems (MIXDES) Conference*, A. Napieralski, Ed., 2011, pp. 684–689.

[21] C. A. Monje, B. M. Vinagre, A. J. Calderon, V. Feliu, and Y. Chen, "Auto-tuning of fractional lead-lag compensators," in *Proceedings of the 16th IFAC World Congress*, 2005.

[22] C. Monje, B. Vinagre, V. Feliu, and Y. Chen, "Tuning and auto-tuning of fractional order controllers for industry applications," *Control Engineering Practice*, vol. 16, no. 7, pp. 798–812, 2008.

[23] C. A. Monje, Y. Chen, B. Vinagre, D. Xue, and V. Feliu, *Fractional-order Systems and Controls: Fundamentals and Applications*, ser. Advances in Industrial Control. Springer Verlag, 2010.

[24] W. Chen, *Passive, Active, And Digital Filters*. Taylor & Francis, 2006.

[25] J. A. T. Machado, "Discrete-time fractional-order controllers," *Journal of Fractional Calculus and Applied Analysis*, vol. 4, no. 1, pp. 47–66, 2001.

Publication 4

Reference

A. Tepljakov, E. Petlenkov, and J. Belikov, "Gain and order scheduled fractional-order PID control of fluid level in a multi-tank system," in 2014 International Conference on Fractional Differentiation and its Applications, 2014, pp. 1–6.

Abstract

Gain scheduling is widely regarded as an effective nonlinear control technique, and its extension to fractional-order control is a natural step. In this paper, we investigate a particular method based on a gain and order scheduling approach for fractional-order PID controllers. The method is applied to the control of a real-life laboratory model of an industrial multi-tank system. Gain and order scheduling is realized by means of a control law comprising two static PID controllers and an appropriate control blending rule providing this way means for stability analysis of the control system. The design of controllers for level control in the first tank is carried out by considering linear fractional-order approximations of the nonlinear model of the process with locally applicable frequency-domain robustness specifications. The controller for the second tank is obtained using time-domain optimization of the transient response. In addition, an extended Kalman filter is designed to reduce measurement noise propagation into the control law thereby enhancing the performance of the pump. The majority of necessary computations, including those related to controller design, are performed numerically in the FOMCON toolbox for MATLAB.

Gain and Order Scheduled Fractional-order PID Control Of Fluid Level in a Multi-Tank System

Aleksei Tepljakov, Eduard Petlenkov, and Juri Belikov
Department of Computer Control
Tallinn University of Technology
Ehitajate tee 5, 19086, Tallinn, Estonia
{aleksei.tepljakov, eduard.petlenkov, juri.belikov}@ttu.ee

***Abstract*—Gain scheduling is widely regarded as an effective nonlinear control technique, and its extension to fractional-order control is a natural step. In this paper, we investigate a particular method based on a gain and order scheduling approach for fractional-order PID controllers. The method is applied to the control of a real-life laboratory model of an industrial multi-tank system. Gain and order scheduling is realized by means of a control law comprising two static PID controllers and an appropriate control blending rule providing this way means for stability analysis of the control system. The design of controllers for level control in the first tank is carried out by considering linear fractional-order approximations of the nonlinear model of the process with locally applicable frequency-domain robustness specifications. The controller for the second tank is obtained using time-domain optimization of the transient response. In addition, an extended Kalman filter is designed to reduce measurement noise propagation into the control law thereby enhancing the performance of the pump. The majority of necessary computations, including those related to controller design, are performed numerically in the FOMCON toolbox for MATLAB.**

I. Introduction

The gain scheduling approach is a very popular nonlinear control design method successfully applied in numerous applications [1]. In this paper, we show a way to extend the ideas of gain scheduling by leveraging tools found in fractional-order calculus. The latter has seen a significant increase in interest from the control engineering community [2]. Indeed, having additional flexibility in the form of noninteger operator powers allows to achieve robust controller tuning with respect to a more strict set of specifications [3]. Notable contributions in fractional-order control are described in [4], [5], [6], [7]. Several recent papers also deal with the problem of gain and order scheduling in relation to fractional-order PID (FOPID) controllers, e.g., [8], which is also the main focus of this work.

Fractional-order control is readily applicable in the industry [9]. However, before carrying the designed methods over to the control of real industrial plants, where a design mistake may result in production losses, it is important to evaluate the performance of the controller using a real-life model of the plant of interest. In this paper, we investigate the problem of fluid level control using a laboratory model of a multi-tank system, which can be configured in a variety of ways. This system serves as a model for a wide range of industrial processes [10], such as those found in chemical engineering, food processing, and irrigation systems.

We now outline the particular contribution of the present paper. First, we propose a method for the gain and order scheduling based on the design of a family of FOPID controllers for the control of a given nonlinear system. We then formulate a particular control law comprising two static FOPID controllers. The proposed structure, on one hand, allows to carry out stability analysis of the resulting control system, and on the other hand simplifies its implementation on embedded systems. Next, we study the multi-tank system and identify the parameters of the corresponding nonlinear model. In addition, we implement a nonlinear Kalman filter to tackle the problem of strong measurement noise. We then turn to the issue of level control in the first tank, and with that goal in mind linear fractional-order approximations are identified at two working points of the system, and fractional-order controllers are designed by means of constrained optimization given local robustness criteria derived from frequency-domain analysis. Once the controllers are obtained, stability analysis is done for the composite control law with every linear approximation. A fluid level control strategy is then proposed for the second tank, and a FOPI controller is designed based on the optimization of the transient response of the control system with the gain and order scheduled controller running simultaneously in the simulated loop.

The paper is organized as follows. In Section II we provide an overview of the mathematical tools and methods derived from fractional-order calculus that are used in this work. In Section III the gain and order scheduling method for FOPID controllers is proposed and studied. In Section IV the model of the multi-tank system is presented, results of parameter identification are given, and the application of the extended Kalman filter is described and its impact on the robustness of the system is analyzed. At this point we also put forth the control problem based on a particular configuration of the multi-tank system. Experimental results follow in Section V. Items for discussion are proposed in Section VI and conclusions and future perspectives are drawn in Section VII.

II. Overview of Employed Fractional-Order Calculus Tools

In this work, we use the Grünwald-Letnikov definition of the fractional operator for the purposes of identifying fractional-order linear approximations. In particular, we use a time-domain simulation method described in [2], [7], [11].

978-1-4799-2591-9/14/$31.00 ©2014 IEEE

The definition has the form

$$ {}_a\mathscr{D}_t^{\alpha} f(t) = \lim_{h\to 0} \frac{1}{h^{\alpha}} \sum_{j=0}^{k} (-1)^j \binom{\alpha}{j} f(t-jh), \qquad (1) $$

where $a = 0$, $t = kh$, k is the number of computation steps and h is step size. For implementing real-time control we use the well-established Oustaloup filter method [4] summarized next. For a frequency range (ω_b, ω_h) and of approximation order N the filter of order $2N + 1$ for an operator s^{γ}, $0 < \gamma < 1$, is given by

$$ \begin{aligned} s^{\gamma} &\approx K \prod_{k=-N}^{N} \frac{s+\omega_k'}{s+\omega_k}, \quad K = \omega_h^{\gamma}, \quad \omega_r = \frac{\omega_h}{\omega_b}, \qquad (2) \\ \omega_k' &= \omega_b(\omega_r)^{\frac{k+N+\frac{1}{2}(1-\gamma)}{2N+1}}, \quad \omega_k = \omega_b(\omega_r)^{\frac{k+N+\frac{1}{2}(1+\gamma)}{2N+1}}. \end{aligned} $$

To identify a linear fractional-order model we employ an output error minimization method, fitting the transient response of a fractional-order transfer function to that of the original system under a prescribed excitation signal. That is, the objective function is given by an output error norm $\|e\,(t)\|_2^2$, where $e(t) = y(t) - \tilde{y}(t)$. The general form of the fractional-order transfer function may be taken as

$$ G(s) = \frac{b_m s^{\beta_m} + b_{m-1} s^{\beta_{m-1}} + \cdots + b_0 s^{\beta_0}}{a_n s^{\alpha_n} + a_{n-1} s^{\alpha_{n-1}} + \cdots + a_0 s^{\alpha_0}} \mathrm{e}^{-Ls}, \qquad (3) $$

where, in general, it is assumed that $\alpha_0 = \beta_0 = 0$, which implies a static gain $K = b_0/a_0$, and L is the input-output delay. Identification is accomplished by means of searching for a suitable parameter set $\theta = [\ a_p \quad \alpha_p \quad b_z \quad \beta_z \quad L\ ]$, comprised of factors and exponents of terms in the pole and zero polynomials, respectively, and of the input-output delay parameter. In this work we use the fractional first-order plus dead time model (FO-FOPDT). A model of this type was previously considered and successfully applied to process control problems in, e.g., [12] and in our earlier works [13], [14]. It has the form

$$ G(s) = \frac{K}{1 + Ts^{\alpha}} \mathrm{e}^{-Ls}. \qquad (4) $$

Assuming that the delay L of the system can be neglected, we can reduce the set of search parameters to $\theta = [\ K \quad T \quad \alpha\ ]$.

Stability analysis for a general fractional-order system given by a transfer function is carried out by means of Matignon's theorem [2], [15], which is provided here for convenience.

Theorem 1: (Matignon's stability theorem) The fractional transfer function $G(s) = Z(s)/P(s)$ is stable if and only if the following condition is satisfied in σ-plane:

$$ |\arg(\sigma)| > q\frac{\pi}{2}, \forall \sigma \in C,\ P(\sigma) = 0, \qquad (5) $$

where $\sigma := s^q$. When $\sigma = 0$ is a single root of $P(s)$, the system cannot be stable. For $q = 1$, this is the classical theorem of pole location in the complex plane: no pole is in the closed right plane of the first Riemann sheet.

For control purposes we consider the parallel form of the fractional-order PI$^{\lambda}$D$^{\mu}$ controller given by the transfer function

$$ C(s) = K_p + \frac{K_i}{s^{\lambda}} + K_d s^{\mu} \qquad (6) $$

and a standard negative unity feedback loop.

The tuning of the FOPI and FOPID controllers is done by means of minimizing a suitable performance index. In this work we consider the integral time-absolute error index

$$ ITAE = \int_0^{\tau} t\,|e(t)|\,\mathrm{d}t. $$

To ensure robustness of the control system we employ the following specifications in the frequency-domain [7]:

- Gain margin G_m and phase margin φ_m specifications;
- Complementary sensitivity function $T(j\omega)$ constraint, providing A dB of noise attenuation for frequencies $\omega > \omega_t$ rad/s;
- Sensitivity function $S(j\omega)$ constraint for output disturbance rejection, providing a sensitivity function of B dB for frequencies $\omega < \omega_s$ rad/s;
- Robustness to plant gain variations: a flat phase of the system is desired within a region of the system critical frequency ω_{cg}.

Finally, since we are working with a real-life model, it makes sense to set control law $u(t)$ saturation values such that $u(t) \in [u_{min}, u_{max}]$.

III. The Proposed Gain and Order Scheduling Method for Fractional-order PID Controllers

In the following we provide a summary of the proposed method. Suppose that a nonlinear system is modeled by

$$ \dot{x} = f(x,u), \quad y = h(x). \qquad (7) $$

Suppose in addition, that a linear fractional-order approximations of the form (3) may be obtained for a set of working points $\{(u_k; y_k), k = 1, 2, \ldots, n\}$, across the system operating range. Denote by

$$ \Psi = \{G_1, G_2, \ldots, G_n\} \qquad (8) $$

the set of such linear fractional-order approximations. Then, for each $G_i \in \Psi$ design a controller of the form (6), that would locally satisfy a set of performance specifications, provided in Section II thereby forming another set, denoted by

$$ \Omega = \{C_1, C_2, \ldots, C_n\}. \qquad (9) $$

Now, consider the composite control law

$$ \Upsilon(x,s) = \sum_{k=1}^{n} \beta_k(x) C_k(s), \qquad (10) $$

where $\beta_k(x)$ is a weighting function depending on the scheduled state $x(t)$ and $C_k(s) \in \Omega$.

The choice of n in (10) depends on the operating range of the system in (7). In the following we consider the case $n = 2$. Then,

$$ \Upsilon(x,s) = \beta_1(x) C_1(s) + \beta_2(x) C_2(s) \qquad (11) $$

and since we are dealing with level control, we may choose the state $x(t)$ to be the level, x_{max} the maximum level, and define

$$\beta_1(x) := \frac{(1-\gamma(x))}{2},\ \beta_2(x) := \frac{\gamma(x)}{2},\ \gamma(x) := \frac{x(t)}{x_{max}}. \quad (12)$$

It is obvious, that since each entry in (9) was designed for a particular linear approximation, the composite control law in (10) must be verified across the whole range of linearized models, that is, stability must be ensured for all entries in (8). In this work, we consider a heuristic method. Since we employ the negative unity feedback loop, we may compose a set

$$\Lambda = \{\Gamma_1, \Gamma_2, \ldots, \Gamma_{\nu n}\} \quad (13)$$

where

$$\Gamma_k = \frac{Z_k(s)}{P_k(s)} = \frac{\Upsilon_j(x,s)G_k(s)}{1+\Upsilon_j(x,s)G_k(s)} \quad (14)$$

and $j = 1, 2, \ldots, \nu$ is the number of state values considered for the test and Υ_j is a particular control law. For each entry in (13) take the characteristic polynomial $P_k(s)$, find the commensurate order $q \geqslant q_{min}$ and use Matignon's theorem.

Remark 1: For the case $n = 2$ in (10) using controllers $C_1(s)$ and $C_2(s)$ given by their parameter sets $(K_{p1}, K_{i1}, \lambda_1, K_{d1}, \mu_1)$ and $(K_{p2}, K_{i2}, \lambda_2, K_{d2}, \mu_2)$ we obtain the characteristic polynomial $P(s)$ which depends on the particular parameter set (K, T, α) of the models of the form (4) in Ψ with $L = 0$, and on the function $\gamma(x)$ of the scheduled state x. The characteristic polynomial has the following form

$$P(s) = a_6 s^{\alpha+\lambda_1+\lambda_2} + a_5 s^{\lambda_1+\lambda_2+\mu_2} + a_4 s^{\lambda_1+\lambda_2+\mu_1} + a_3 s^{\lambda_1+\lambda_2} + a_2 s^{\lambda_1} + a_1 s^{\lambda_2}, \quad (15)$$

where $a_6 = 2T$, $a_5 = K \cdot K_{d2}\gamma(x)$, $a_4 = K \cdot K_{d1}(1-\gamma(x))$, $a_3 = 2 + K \cdot K_{p1}(1-\gamma(x)) + K \cdot K_{p2}\gamma(x)$, $a_2 = K \cdot K_{i2}\gamma(x)$, and $a_1 = K \cdot K_{i1}(1-\gamma(x))$. The stability test works with commensurate-order systems. When the resulting fractional-order system is not of commensurate order, the stability test produces approximate results [2].

Remark 2: It is important to stress, that the controllers in Ω have static parameters and operate simultaneously, while the scheduling, that is the choice of the control action, is done by means of blending functions. Using a static description of the controllers should improve reliability of embedded control implementations [16]. Online gain and order scheduling is possible.

IV. Model of the Multi-Tank System

In this section we describe the particular configuration of the multi-tank system [17] and state the control problem.

A. General Description

The multi-tank system consists of three distinctly shaped tanks with level sensors and mechanical and automatic valves, a water reservoir, and a pump, that connects the reservoir and the upper tank. In this work we consider level control in the first two tanks, that is in the upper one and the middle one. An illustration is provided in Fig. 1.

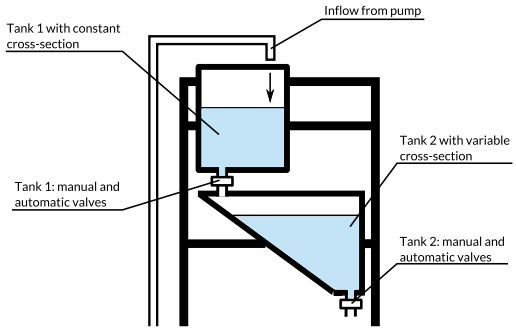


Fig. 1. Configuration of the multi-tank system

This system can be described by the following differential equations:

$$\dot{x}_1 = \frac{1}{\eta_1(x_1)}\left(u_p(v) - C_1 x_1^{\alpha_1} - \zeta_1(v_1)x_1^{\alpha_{v1}}\right), \quad (16)$$
$$\dot{x}_2 = \frac{1}{\eta_2(x_2)}\left(q + r - C_2 x_2^{\alpha_2} - \zeta_2(v_2)x_2^{\alpha_{v2}}\right),$$

where x_1 and x_2 are levels in the upper tank and middle tank, respectively, $\eta_1(x_1) = A = aw$ is the constant cross-sectional area of the upper tank, $\eta_2(x_2) = cw + x_2 bw/x_{2max}$ is the variable cross-sectional area of the middle tank, $u_p(v)$ is the pump capacity that depends on the normalized control input $v(t) \in [0,1]$, $\zeta_1(v_1)$ and $\zeta_2(v_2)$ are variable flow coefficients of the automatic valves controlled by normalized input signals $v_1(t), v_2(t) \in [0,1]$, $q = C_1 x_1^{\alpha_1}$ and $r = \zeta_1(v_1)x_1^{\alpha_{v1}}$. Fluid levels are taken as outputs of the system, i.e. $y_1 = x_1$, $y_2 = x_2$. A list of parameters of the model with their respective physical meaning is provided in Table I.

TABLE I. Parameters of the two-tank system

Parameter	Physical description
w	width of both tanks
a	length of the upper tank
b, c	lengths of the top and bottom part of the middle tank
x_{2max}	maximum attainable fluid level in the middle tank
C_i	resistance of the output orifice of the ith tank
α_i, α_{vi}	state exponents describing the flow of the ith valve

The multi-tank system may be controlled by means of a personal computer running MATLAB and Simulink software packages. The identification of parameters above is done by means of a series of experiments. The validation of the model in (16) was carried out under a variety of excitation signals. It can be seen from Fig. 2 that a sufficiently accurate model is obtained.

B. Statement of the Control Problem

The task is to design a controller for the upper tank such that would keep the level of fluid within reasonable bounds at the desired set point in the presense of disturbances caused by the controlled output valve. Also it is required to design a controller for the middle tank, such that would keep the level

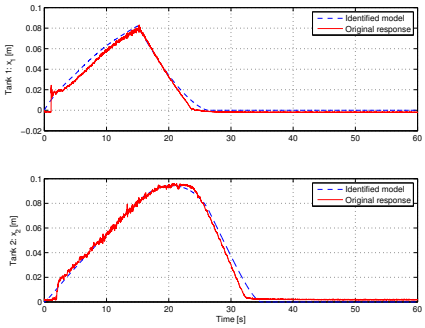


Fig. 2. Model validation results

of fluid at the desired set point using controlled valves of the upper tank and also its own valve. To this end, we will now define a unified control input for controlling the level in the second tank $v_c(t) \in [-1, 1]$ such, that the control inputs of the automatic valves are given by the following set of rules

$$v_1 = \begin{cases} 0, & \text{if } v_c \leqslant 0, \\ 0.3v_c + v_d, & \text{if } v_c > 0, \end{cases} \tag{17}$$

and

$$v_2 = \begin{cases} 0, & \text{if } v_c \geqslant 0, \\ -0.3v_c + v_d, & \text{if } v_c < 0. \end{cases} \tag{18}$$

The value $v_d = 0.7$ corresponds to the deadzone of the control in both cases, that is, the fluid does not flow through the automatic valves when $v_1 \leqslant v_d$ or $v_2 \leqslant v_d$. The constructed control law allows to regulate the fluid level in the middle tank. The tanks are, in fact, coupled, so only a limited range of fluid level values is achievable in the middle tank and it is related to the level in the upper tank. The outflow of liquid from the upper tank through the automatic valve forms part of the control for the middle tank and is considered a disturbance from the perspective of level control in the upper tank.

C. Derivation of the Extended Kalman Filter

One significant problem with the present implementation of the multi-tank system is the amount of noise present in level measurements. Since the ends of the submerged sensor tubes are placed very close to the output valves in all three tanks, the switching of the automatic valves creates additional noise which cannot be easily dealt with using linear filtering methods alone. Therefore, it was decided to employ an extended Kalman filter [18] to tackle this problem. To conserve space we provide here only the final equations with a brief summary.

In essence, the purpose of the extended Kalman filter is to estimate unmeasured states and actual process outputs using statistical methods. This is done in two major steps: the predictor step, and the corrector step. We can construct an individual Kalman filter for each of the tanks in the system. This allows us to operate with scalar values in the filter equations and hence computations are considerably simplified. We use a discrete-time nonlinear model of a single tank obtained by the Euler method

$$\begin{aligned} x_k &= x_{k-1} + \frac{T_s}{\eta(x)} u_{k-1} - \frac{CT_s}{\eta(x)} x_{k-1}^{\alpha} \\ y_k &= x_k, \end{aligned} \tag{19}$$

where $\eta(x)$ is the cross-sectional area which depends on the level x in the tank, T is the sampling interval, C and α together form the outflow characteristic, and u is the inflow, k is the discrete sample index. Then, we construct the equations for the discrete-time extended Kalman filter based on the nonlinear model of the process and formed by the predictor step

$$\hat{x}_k^- = \hat{x}_{k-1} + \left(\frac{T_s}{\eta(\hat{x})}\right) u_k - \left(\frac{CT_s}{\eta(\hat{x})}\right) \hat{x}_{k-1}^{\alpha}, \tag{20}$$

$$P_k^- = P_{k-1} \left(1 - \left(\frac{CT_s}{\eta(\hat{x})}\right) \alpha \hat{x}_{k-1}^{(\alpha-1)}\right)^2 + Q, \tag{21}$$

and the corrector step

$$K_k = \frac{P_k^-}{P_k^- + R} \tag{22}$$

$$\hat{x}_k = \hat{x}_k^- + K_k \left(\tilde{y}_k - \hat{x}_k^-\right) \tag{23}$$

$$P_k = (1 - K_k) P_k^-, \tag{24}$$

where Q and R are the filter design variables representing input and measurement noise covariances, respectively. We have found that $Q = 10^{-5}$ and $R = 10^{-2}$ provide the necessary performance.

Note, that we neglect the automatic valve in this model. In Fig. 3 we provide the estimation error for the upper tank only. At time $t = 20$s the automatic valve leading from the upper tank to the middle one is fully opened. It is possible to observe the shift of the estimation mean, which is drawn over the noisy error signal for reference. The estimation error mean $\bar{e}$ reaches the maximum value of $\bar{e}_{max} = 0.0012$m which is within the control error bounds.

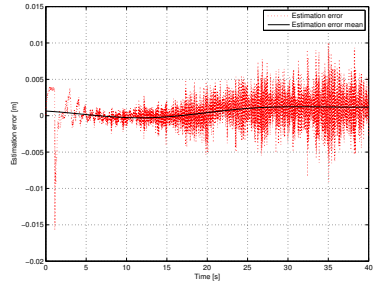


Fig. 3. EKF estimation error for the upper tank

V. Experimental Results

We now illustrate the use of the method proposed in Section III to the problem of level control in the two tanks. Numerical analysis is done by means of the FOMCON toolbox for MATLAB [19], [20]. The real-life system used in the experiments is depicted in Fig. 4.

First, linear approximations of the form (4) are obtained from the nonlinear model by means of time-domain identification at system working points $(0.7029, 0.1)$ and $(0.7879, 0.2)$.

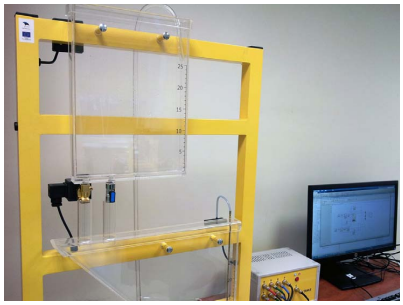


Fig. 4. Real-life multi-tank system

The identification procedure is detailed in [20]. The following models are found:

$$G_1(s) = \frac{0.14464}{18.728s^{0.91746} + 1}$$

and

$$G_2(s) = \frac{0.25881}{27.859s^{0.9115} + 1}.$$

During our study we have found that the fractional exponent α in the models above frequently arises during the identification process, and more importantly, these fractional models offer superior precision than classical ones. With variation of certain model parameters we have identified a range that the exponent α can take to be approximately $\alpha \in (0.88, 0.94)$. This result may, in fact, be related to the one in [13]. However, no clear relation has been identified between the fractional exponents of the process models and those in the states of the nonlinear model in (16).

Next, controllers are designed for level control in the upper tank using the FOPID optimization tool of FOMCON toolbox. For this a nonlinear model of the system is used for simulations in the time domain, the set value corresponds to the particular operating point. Linear fractional-order approximations corresponding to the working points are used to constrain the optimization process by employing frequency-domain specifications [3].

In particular, in case of the first controller, a phase margin is set to $\varphi_m \geqslant 60°$, sensitivity and complementary sensitivity function constraints are set such that $\omega_t = 0.02$ and $\omega_s = 0.1$ with $A = B = -20$ dB. Robustness to gain variations specification is also used with the critical frequency $\omega_c = 0.1$. For the second controller, the phase margin specification is changed to $\varphi_m = 85°$ and the bandwidth limitation specified by ω_c is removed. Due to the flexibility of the tuning tool, it is possible to retune the controllers by considering the composite control law in (11) during the controller optimization process.

Finally, two FOPID controllers are obtained:

$$C_1(s) = 6.1467 + \frac{1.0712}{s^{0.9528}} + 0.8497s^{0.8936}$$

and

$$C_2(s) = 5.1524 + \frac{0.3227}{s^{1.0554}} + 2.4827s^{0.010722}.$$

The composite control law

$$C(s) = \frac{(1 - \gamma(x_1))\, C_1(s) + \gamma(x_1) C_2(s)}{2}$$

is then verified with both models $G_1(s)$ and $G_2(s)$ using (15) with step size of $\gamma(x_1) = 0.01$ and minimum commensurate order $q_{min} = 0.01$. The result of the test is that the closed-loop systems are stable in case of both fractional models.

Once the gain and order scheduled composite controller is designed, it is plugged into the control system, and a FOPID controller is designed for the control of the level in the second tank using the same optimization tool of the FOMCON toolbox. Here we need to consider the following. First, frequency-domain specifications are not applicable, since we do not have a linear model of this process. In addition, the application of the D$^\mu$ component is not very desirable in this case, because of the relatively large dead-zone in control of the outflow and therefore any noise amplified due to the differentiator component will lead to rapid switching of the automatic valves.

Therefore we design a FOPI controller based only on optimization of the transient response of the control system in the time domain. The following controller is obtained:

$$C_3(s) = 5.0000 + \frac{0.06081}{s^{0.1029}}$$

which is essentially a proportional controller with a weak fractional-order integrator.

Finally, the performance of the whole control system is evaluated with the real-life plant. All controllers are implemented by means of (2). A Simulink model is used, which is given in Fig. 5. A low-pass filter is added to the level control loop of the second tank. This introduces a phase lag that reduces switching of the automatic valves. Set-points changes are presented in Table II. The result of the performance evaluation is presented in Fig. 6.

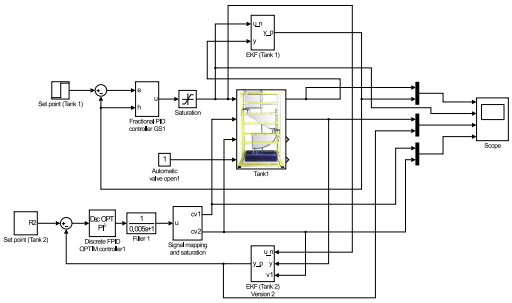


Fig. 5. Simulink diagram used for the real-time control experiment

The levels in the first and second tanks are kept within reasonable error bounds of approximately 5% of the set point. The control task put forth in Section IV is thereby accomplished. Notice, that the noise does not propagate into the control law of the pump. The control law of the valves is, for the most part, bang-bang due to the large deadzone in the control of the automatic valves. However, the switching frequency is acceptable.

TABLE II. REFERNCE SIGNAL CHANGES CONSIDERED FOR THE REAL-TIME CONTROL EXPERIMENT

Time instance [s]	Upper tank reference [m]	Middle tank reference [m]
0	0.1	0.1
100	0.15	0.1
150	0.15	0.2
250	0.2	0.2

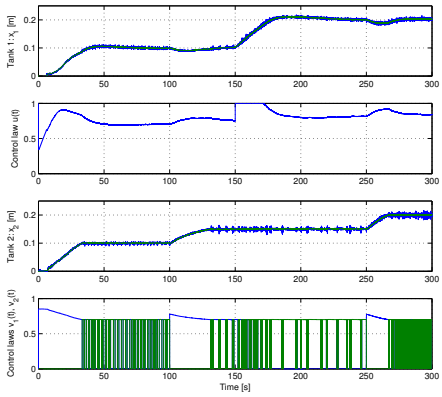


Fig. 6. Control system evaluation results with the real-life plant

VI. DISCUSSION

In the paper, we have presented a flexible gain scheduling method applied to design of control systems comprised of FOPID controllers. Several discussion items are outlined next.

In this work we perform only heuristic linear stability analysis of the resulting composite control system, but it would be more beneficial to consider stability analysis of the nonlinear system. In addition, further work may be carried out to design a more efficient controller for the middle tank, such that would minimize the switching of automatic valves.

The proposed method is quite simple, requires only static description of the FOPID controllers and therefore may be employed in, e.g., automatic tuning for efficient control of nonlinear systems with across a large operating range. This also serves as a good foundation for developing embedded control hardware, which also forms an important part of our future work.

VII. CONCLUSIONS

In this paper, we have presented initial results in relation to an efficient control method involving a composite control law comprised of fractional-order PID controllers applied to the problem of level control in a multi-tank system. The proposed method was successfully applied to the control problem, and relevant results were presented and analyzed.

ACKNOWLEDGMENTS

This work was partially supported by the Estonian Doctoral School in Information and Communication Technology through the interdisciplinary project FOMCON.

REFERENCES

[1] W. Leithead, "Survey of gain-scheduling analysis design," *International Journal of Control*, vol. 73, pp. 1001–1025, 1999.

[2] Y. Q. Chen, I. Petráš, and D. Xue, "Fractional order control - a tutorial," in *Proc. ACC '09. American Control Conference*, 2009, pp. 1397–1411.

[3] A. Tepljakov, E. Petlenkov, and J. Belikov, "A flexible MATLAB tool for optimal fractional-order PID controller design subject to specifications," in *Proceedings of the 31st Chinese Control Conference*, W. Li and Q. Zhao, Eds., Hefei, Anhui, China, 2012, pp. 4698–4703.

[4] A. Oustaloup, P. Melchior, P. Lanusse, O. Cois, and F. Dancla, "The CRONE toolbox for Matlab," in *Proc. IEEE Int. Symp. Computer-Aided Control System Design CACSD 2000*, 2000, pp. 190–195.

[5] I. Podlubny, L. Dorčák, and I. Kostial, "On fractional derivatives, fractional-order dynamic systems and $PI^{\lambda}D^{\mu}$-controllers," in *Proc. 36th IEEE Conf. Decision and Control*, vol. 5, 1997, pp. 4985–4990.

[6] I. Podlubny, "Fractional-order systems and $PI^{\lambda}D^{\mu}$-controllers," *IEEE Trans. Autom. Control*, vol. 44, no. 1, pp. 208–214, 1999.

[7] C. A. Monje, Y. Q. Chen, B. M. Vinagre, D. Xue, and V. Feliu, *Fractional-order Systems and Controls: Fundamentals and Applications*, ser. Advances in Industrial Control. Springer Verlag, 2010.

[8] I. Tejado, B. M. Vinagre, and Y. Q. Chen, "Fractional gain and order scheduling controller for networked control systems with variable delay. application to a smart wheel," *Proceedings of the 4th IFAC Workshop on Fractional Differentiation and Its Applications, University of Extremadura, Badajoz, Spain*, 2010.

[9] C. A. Monje, B. M. Vinagre, V. Feliu, and Y. Q. Chen, "Tuning and auto-tuning of fractional order controllers for industry applications," *Control Engineering Practice*, vol. 16, no. 7, pp. 798–812, 2008.

[10] J. Belikov, S. Nõmm, E. Petlenkov, and K. Vassiljeva, "Application of neural networks based SANARX model for identification and control liquid level tank system," in *The 12th International Conference on Machine Learning and Applications*, Miami, FL, USA, December 2013, pp. 246–251.

[11] D. Xue, Y. Q. Chen, and D. P. Atherton, *Linear Feedback Control: Analysis and Design with MATLAB (Advances in Design and Control)*, 1st ed. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2008.

[12] F. Padula, R. Vilanova, and A. Visioli, "H_{∞} model matching PID design for fractional FOPDT systems," in *American Control Conference (ACC), 2012*, 2012, pp. 5513–5518.

[13] A. Tepljakov, E. Petlenkov, J. Belikov, and M. Halás, "Design and implementation of fractional-order PID controllers for a fluid tank system," in *Proc. 2013 American Control Conference (ACC)*, Washington DC, USA, June 2013, pp. 1780–1785.

[14] A. Tepljakov, E. Petlenkov, J. Belikov, and J. Finajev, "Fractional-order controller design and digital implementation using FOMCON toolbox for MATLAB," in *Proc. of the 2013 IEEE Multi-Conference on Systems and Control conference*, 2013, pp. 340–345.

[15] D. Matignon, "Generalized fractional differential and difference equations: Stability properties and modeling issues," in *Proc. of Math. Theory of Networks and Systems Symposium*, 1998, pp. 503–506.

[16] A. Tepljakov, E. Petlenkov, and J. Belikov, "Synthesis of digital filters for embedded fractional-order control applications," in *Proc. of the Seventh IKTDK Information and Communication technology Doctoral School Conf.*, 2013, pp. 93–96.

[17] (2013) Official website of INTECO Sp. z o. o. [Last access time: 11.03.2013]. [Online]. Available: http://www.inteco.com.pl/

[18] D. Simon, *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*. Wiley, 2006.

[19] A. Tepljakov, E. Petlenkov, and J. Belikov. (2011) FOMCON toolbox. [Online]. Available: http://www.fomcon.net/

[20] A. Tepljakov, E. Petlenkov, and J. Belikov, "FOMCON: a MATLAB toolbox for fractional-order system identification and control," *International Journal of Microelectronics and Computer Science*, vol. 2, no. 2, pp. 51–62, 2011.

Publication 5

Reference

A. Tepljakov, E. Petlenkov, and J. Belikov, "Fractional-order digital filter approximation method for embedded control applications," International Journal of Microelectronics and Computer Science, vol. 5, no. 2, pp. 54–60, 2014.

Abstract

Fractional-order calculus presents a novel modeling approach for systems with extraordinary dynamical properties by introducing the notions of derivatives and integrals of noninteger order. In system theory this gives rise to extensions to linear, time invariant systems to enhance the description of complex phenomena involving memory or hereditary properties of systems. Standard industrial controllers, such as the PID controller and lead-lag compensator, have also been updated to benefit from the effects of noninteger integration and differentiation, and have advantages over classical controllers in case of both conventional and fractional-order process control. However, given the definitions of fractional operators, accurate digital implementation of fractional-order systems and controllers is difficult because it requires infinite memory. In this work we study the digital implementation of a fractional-order PID controller based on an infinite impulse response (IIR) filter structure obtained by applying the Oustaloup recursive filter generation technique. Software for generating digital fractional-order is developed and tested on an Atmel AVR microcontroller. The results are verified using a MATLAB/Simulink based real-time prototyping platform.

Fractional-order Digital Filter Approximation Method for Embedded Control Applications

Aleksei Tepljakov, Eduard Petlenkov, and Juri Belikov

***Abstract*—Fractional-order calculus presents a novel modeling approach for systems with extraordinary dynamical properties by introducing the notions of derivatives and integrals of noninteger order. In system theory this gives rise to extensions to linear, time invariant systems to enhance the description of complex phenomena involving memory or hereditary properties of systems. Standard industrial controllers, such as the PID controller and lead-lag compensator, have also been updated to benefit from the effects of noninteger integration and differentiation, and have advantages over classical controllers in case of both conventional and fractional-order process control. However, given the definitions of fractional operators, accurate digital implementation of fractional-order systems and controllers is difficult because it requires infinite memory. In this work we study the specific implementation of a fractional-order PID controller and fractional-order lead-lag compensator based on an infinite impulse response (IIR) filter structure obtained by applying the Oustaloup recursive filter synthesis technique. Software for generating digital fractional-order is developed and tested on an Atmel AVR microcontroller. The results are verified using a MATLAB/Simulink based real-time prototyping platform.**

***Index Terms*—fractional-order calculus, fractional-order pid control, digital control, embedded system**

I. Introduction

DURING the last 300 years fractional-order calculus has been the subject of moderately active discussion [1]. While the related mathematical theory still has some issues, it is already possible to benefit from additional modeling possibilities, offered by fractional calculus, in terms of applications. Fractional-order control is a notable example, where fractional-order calculus offers numerous opportunities for enhancing the characteristics and performance of control loops.

In the vast field of process control PID controllers stand out as the most popular choice for control loop design. An abundance of tuning rules and techniques has been developed for these truly ubiquitous controllers. Fractional-order PID controllers, introduced in [2], are a natural extension to their conventional counterparts and have superior characteristics due to the added flexibility of the noninteger operators involved in the computation of the control law. If properly tuned, FOPID controllers are capable of outperforming conventional PID controllers in case of both integer-order and fractional-order processes [3], and in particular, in case of servo system control [4], [5], [6]. The implementation of these controllers, however, is not as trivial as in case of classical PID controllers and requires careful consideration, especially when embedded system realizations are concerned.

In this work we treat the case of digital filter synthesis for embedded control applications. The particular contribution of this paper is as follows. We use the well-established Oustaloup filter [7] as basis for obtaining continuous-time zeros and poles, and then apply a discrete-time transformation yielding the digital filter. Further, the zero-pole representation is converted to the second-order section form to ensure computational stability of the resulting filter. While all these operations are easily carried out in MATLAB, specific considerations need to be taken into account when dealing with embedded realization of the digital filter. In particular, we provide equations that may be used to carry out the necessary computations and seek to establish some timing and memory constraints related to the digital implementation of a FOPID controller. To make the method more widely applicable we choose relatively low-end hardware as basis for implementation. We develop the fractional-order PID controller software for the Atmel AVR ATmega8A microcontroller, and test it by means of a MATLAB/Simulink based real-time simulation platform [8]. To validate the resulting embedded control system a coupled tank plant model is used in Simulink for which a fractional-order controller was designed in our earlier work [9]. In addition, we use a model of a position servo from [10] to validate the performance of a fractional lead compensator. In both cases, the modeled plant is connected to the controller prototype by means of a data acquisition board thereby simulating a real-life physical process. Further, this work complements earlier studies [11] and our work [12], [13], [14], where we used FOMCON toolbox for MATLAB [15] to generate the digital filter coefficients and hard-coded them into the microcontroller firmware. The results of this work can be applied to industrial process control.

The paper is organized as follows. In Section II the reader is introduced to the core concepts of fractional-order calculus in modeling and control. In Section III the proposed synthesis method is described. In Section IV the method is carried over to an embedded system realization. The specifics of the implementation are provided. Dynamic memory requirements for the computing device, e.g. a microcontroller, are also analyzed. In Section V the realization of a digital fractional-order PID controller and a digital fractional-order lead-lag compensator is discussed. Real-time simulation results follow in Section VI. Finally, in Section VII conclusions are drawn.

A. Tepljakov, E. Petlenkov and J. Belikov are with the Department of Computer Control, Tallinn University of Technology, Ehitajate tee 5, 19086, Tallinn, Estonia (e-mails: {aleksei.tepljakov, eduard.petlenkov, juri.belikov}@ttu.ee)

This work was partially supported by the Tiger University Program of the Information Technology Foundation for Education and the Estonian Doctoral School in Information and Communication Technology through the interdisciplinary project FOMCON.

Copyright © 2014 by Department of Microelectronics & Computer Science, Lodz University of Technology

II. Overview of Fractional-Order Calculus Applications to Modeling and Control

The cornerstone of fractional-order modeling is the generalized non-integer order fundamental operator

$$ {}_a\mathscr{D}_t^{\alpha} = \begin{cases} \mathrm{d}^{\alpha}/\mathrm{d}t^{\alpha} & \alpha > 0, \\ 1 & \alpha = 0, \\ \int_a^t (\mathrm{d}\tau)^{-\alpha} & \alpha < 0, \end{cases} \tag{1} $$

where a and t denote the limits of the operation. There exist several definitions of the integro-differential operator. We consider the Grünwald-Letnikov definition [1], [10] due to its immediate applicability to numeric evaluation of fractional-order derivatives. The definition is as follows:

$$ {}_a\mathscr{D}_t^{\alpha} f(t) = \lim_{h \to 0} \frac{1}{h^{\alpha}} \sum_{j=0}^{k} (-1)^j \binom{\alpha}{j} f(t - jh), \tag{2} $$

where $a = 0$, $t = kh$, k is the number of computation steps and h is step size. We notice, that this definition is not directly suitable for real-time signal processing applications, since infinite sample memory is required [1], [16].

If zero initial conditions are assumed, the Laplace transform of the fractional α-order derivative is given by

$$ \int_0^{\infty} \mathrm{e}^{-st} {}_0\mathscr{D}_t^{\alpha} f(t) \mathrm{d}t = s^{\alpha} F(s), \tag{3} $$

where $\alpha \in \mathbb{R}^+$ and s is the usual Laplace transform variable. The parallel form of the fractional PID controller, called the PI$^{\lambda}$D$^{\mu}$ controller with λ and μ being the fractional orders of the integral and differential components has the following transfer function:

$$ C(s) = K_p + \frac{K_i}{s^{\lambda}} + K_d \cdot s^{\mu}. \tag{4} $$

It can be seen, that in the frequency domain this controller offers more tuning flexibility since more freedom in changing the shape of the response is achieved. Because we are dealing with band-limited approximations throughout this work, it is important to implement the fractional-order integrator component in (4) as

$$ G_I(s) = \frac{1}{s^{\lambda}} = \frac{s^{1-\lambda}}{s} \tag{5} $$

for $\lambda < 1$ since this ensures the effect of an integer-order integrator at low frequencies thereby resulting in faster convergence of the controlled output to its final value [10].

Consider also a particular type of fractional controller, namely, the fractional lead-lag compensator. It has the following form:

$$ G_c(s) = K_c x^{\alpha} \left(\frac{\lambda s + 1}{x\lambda s + 1} \right)^{\alpha}, 0 < x < 1, \tag{6} $$

where α is the fractional order of the controller, $1/\lambda = \omega_z$ is the zero frequency and $1/(x\lambda) = \omega_p$ is the pole frequency when $\alpha > 0$. Parameters K_c, x, α and λ are the design parameters.

III. The Discrete-time Oustaloup Filter Generation Method

Let us first summarize the original Oustaloup method [7]. Given the approximation frequency range $[\omega_b, \omega_h]$ rad/s, order of approximation $\nu \in \mathbb{Z}^+$ and fractional power $\alpha \in [-1, 1] \subset \mathbb{R}$, we proceed to compute $(2\nu + 1)$ zeros and $(2\nu + 1)$ poles of the filter as

$$ \omega'_k = \omega_b \theta^{\frac{(k+\nu+0.5-0.5\alpha)}{2\nu+1}}, \quad \omega_k = \omega_b \theta^{\frac{(k+\nu+0.5+0.5\alpha)}{2\nu+1}}, \tag{7} $$

where $k = \{-\nu, -\nu+1, \ldots, 0, \ldots, \nu-1, \nu\}$ and $\theta = \omega_h/\omega_b$. Thus the continuous recursive Oustaloup filter transfer function is obtained in the form

$$ \hat{G}(s) = \omega_h^{\alpha} \frac{(s - \omega'_{-\nu})(s - \omega'_{-\nu+1}) \cdots (s - \omega'_{\nu})}{(s - \omega_{-\nu})(s - \omega_{-\nu+1}) \cdots (s - \omega_{\nu})}. \tag{8} $$

The filter approximates a fractional-order operator

$$ s^{\alpha} \approx \hat{G}(s) \tag{9} $$

in the chosen frequency range. The amount of ripple in the phase response of this filter can also be determined. In the following, we describe the discretization method which, when employed, can serve as a basis for discrete-time Oustaloup filter generation and may be implemented on an embedded device.

Suppose that we are given a sampling interval $T_s \in \mathbb{R}^+$. Then we may set the higher frequency bound of approximation in (7) to $\omega_h = 2/T_s$. Next, consider the zero-pole matching equivalents method for obtaining a discrete-time equivalent of a continuous time transfer function [17]. The following mapping is used for both zeros and poles:

$$ z = \mathrm{e}^{sT_s}, \tag{10} $$

where s denotes a particular zero or pole. Therefore, for each k in (7) we take

$$ \sigma'_k = \mathrm{e}^{-T_s \omega'_k}, \quad \sigma_k = \mathrm{e}^{-T_s \omega_k} \tag{11} $$

thus mapping continuous zeros and poles to their discrete-time equivalents directly. We notice, that once the mapping is done, we need to compute the gain of the resulting discrete-time system at the central frequency $\omega_u = \sqrt{\omega_b \omega_h}$. This can be done by first finding the gain of the resulting discrete-time system by taking

$$ K_u = \left| H(\mathrm{e}^{j\omega_u T_s}) \right|. \tag{12} $$

We also know the correct gain at this frequency

$$ K_s = \omega_u^{\alpha}. \tag{13} $$

So, finally we obtain the gain of the system as

$$ K_c = K_s / K_u. \tag{14} $$

The discrete-time system is thus described by a transfer function of the form

$$ H(z) = K_c \frac{(z - \sigma'_{-\nu})(z - \sigma'_{-\nu+1}) \cdots (z - \sigma'_{\nu})}{(z - \sigma_{-\nu})(z - \sigma_{-\nu+1}) \cdots (z - \sigma_{\nu})}. \tag{15} $$

Due to the fact that the order of the approximated operator is $\alpha \in \mathbb{R}^+$ the transfer function in (8) is stable [7] and the corresponding discrete-time equivalent (15) is also stable.

IV. Considerations for Embedded Device Implementation

In this section we would like to address the problems associated with implementing the generation scheme described above on an embedded device, such as a microcontroller. We have to take the following into consideration:

- Performance limitations;
- Limited computational abilities;
- Potential memory size limitations.

The first item completely depends on the type of microprocessor (and potentially additional hardware computational units) used in the implementation.

We notice, that (12) involves computations with complex numbers. However, we can compute a particular factor $(z-\sigma)$ in (15) at the frequency ω_u as follows

$$\left|e^{j\omega_u T_s}-\sigma\right| = \left|\cos(\omega_u T_s)+j\sin(\omega_u T_s)-\sigma\right| = \\ = \sqrt{1-2\sigma\cos(\omega_u T_s)+\sigma^2} \quad (16)$$

due to Euler's formula. Therefore, the gain of the system specified by discrete-time zeros and poles in (11) may be computed as

$$K_u = \frac{\prod_{k=-\nu}^{\nu}\left(1-\sigma'_k\theta+(\sigma'_k)^2\right)^{0.5}}{\prod_{k=-\nu}^{\nu}\left(1-\sigma_k\theta+\sigma_k^2\right)^{0.5}}, \quad (17)$$

where $\theta = 2\cdot\cos(\omega_u T_s)$ is constant at the given frequency ω_u and sampling interval T_s which needs to be computed only once. After computing this gain one arrives at the final gain K_c of the discrete-time approximation by using equation (14).

This system can be implemented as an IIR filter. The next step is to transform this representation into second-order section form to improve computational stability. Consider the set of discrete-time zeros (poles), that we have obtained earlier

$$z = \{\sigma_{-\nu}, \sigma_{-\nu+1}, \ldots, \sigma_0, \ldots \sigma_\nu, \sigma_{\nu-1}, \sigma_\nu\}. \quad (18)$$

Due to the generation method (7) the set in (18) is an ordered set. In order to arrive at the second-order section form for the zero (pole) polynomial we proceed as follows. We have $2\nu+1$ zeros (poles), so there are $\nu+1$ second-order sections (including a single first-order section). Therefore, we have the polynomial

$$h(z) \quad = \quad (1-\sigma_\nu z^{-1})\cdot\prod_{k=0}^{\nu-1}\zeta(z) \quad (19)$$

in the variable z, where $\zeta(z) = 1+(c_k+d_k)z^{-1}+(c_k\cdot d_k)z^{-2}$, $c_k = -\sigma_{-\nu+2k}$ and $d_k = -\sigma_{-\nu+2k+1}$. So finally we arrive at the form

$$H(z) = K_c\prod_{k=1}^{\nu}\frac{1+b_{0k}z^{-1}+b_{1k}z^{-2}}{1+a_{0k}z^{-1}+a_{1k}z^{-2}}, \quad (20)$$

which can be effectively used as an IIR filter in control applications.

We now turn to the issue of storing the aforementioned coefficients on an embedded system with dynamic memory. Additionally we consider the necessary memory size for digital signal processing related computations.

While it is usually possible to use dynamic memory allocation for both zero/pole generation and SOS coefficient arrays, on embedded systems with limited memory size it is safer to use a static memory allocation architecture to circumvent potential run-time problems arising from, e.g., memory fragmentation. Therefore, care must be taken to choose a sufficient maximal approximation order ν_{max} and preallocate the necessary array memory space beforehand. Suppose that a floating-point data type with a size of ψ bytes is available in a particular implementation. In the following, we provide some computations related to minimal memory requirements. First, to store arrays of values for discrete zero/pole calculation:

$$\text{Memory for zero/pole arrays} = 2\psi(2\nu_{max}+1)\,\text{bytes}. \quad (21)$$

Now we provide the memory requirements for second-order section coefficient storage. Note, that in (20) we only need to store coefficients b_1, b_2, a_1 and a_2. Then for both arrays we have

$$\text{Memory for SOS arrays} = 4\psi\chi(\nu_{max}+1)\,\text{bytes}, \quad (22)$$

where χ is the number of approximated operators. Finally, we will need memory elements for the digital signal processing application:

$$\text{Memory for DSP} = 2\psi\chi(\nu_{\max}+1)\,\text{bytes}.$$

The total amount of memory required for the arrays is thus

$$\text{Total memory} = 2\psi\left((3\chi+2)\nu_{max}+3\chi+1\right)\ \text{bytes}. \quad (23)$$

Consider, for example, the Atmel AVR ATmega8 microcontroller, which we use as the basis for the implementation of a digital filter approximating a fractional-order operator with an order α such that $0<\alpha<1$. Suppose that a single precision floating-point data type is available. Then $\psi = 4$ and $\chi = 1$ and for $\nu_{max} = 10$ we need to preallocate

$$2\cdot 4\left((3\cdot 1+2)\cdot 10+3\cdot 1+1\right) = 432\,\text{bytes},$$

which takes up 42.18% out of 1024 bytes of SRAM memory of this particular microcontroller.

We remark, that it is possible to reuse some static memory blocks during the generation of the coefficients thus reducing the necessary memory size requirements.

V. Digital Realization of Fractional-order Controllers

In the following section we summarize the process of digital implementation of a fractional-order PI$^\lambda$D$^\mu$ controller and a fractional lead-lag compensator.

A. Fractional-order PID Controller

We may digitally implement the fractional-order PID controller as

$$H_{PI^\lambda D^\mu}(z) = K_p + K_i H_I^{-\lambda}(z) + K_d H_D^\mu(z^{-1}), \quad (24)$$

where K_p, K_i, and K_d are gains of the parallel form of the controller as in (4), $H_I^{\lambda}(z)$ corresponds to a discrete-time approximation of a fractional-order integrator of order λ and $H_D^{\mu}(z)$ corresponds to a discrete-time approximation of a fractional-order differentiator of order μ, such that $\lambda, \mu \in [0, 1]$.

Next, we address the issue of implementing the fractional-order integrator. Recall the discussion in Section II. Due to (5) we should implement the integrator as

$$H_I(z) = H^{1-\lambda}(z) \cdot H_I(z), \tag{25}$$

where $H^{1-\lambda}(z)$ is computed using the method presented above, and

$$H_I(z) = \frac{T_s}{(1 - z^{-1})} \tag{26}$$

is the discrete-time integrator.

B. Fractional-order Lead-Lag Compensator

A fractional lead-lag compensator in (6) has the following simplified form:

$$C_L(s) = K_L \left(\frac{b_L s + 1}{a_L s + 1} \right)^{\alpha_L}, \tag{27}$$

where K_L, b_L, a_L, and α_L are design parameters. To implement this controller one must choose the appropriate approximation frequency bounds ω_b and ω_h in (7) such that

$$\omega_b = 1/b_L, \quad \omega_h = 1/a_L. \tag{28}$$

In addition, a correction gain $K_c = b_L^{\alpha}$ must be applied to the Oustaloup filter approximation. The approximation is then given by

$$\hat{C}_L(s) = K_L K_c \hat{G}(s), \tag{29}$$

where $\hat{G}(s)$ is computed in (8). It can be easily deduced, that a fractional lag compensator corresponds to a I^{λ} controller with

$$K_i = K_L K_c, \quad \lambda = \alpha_L \tag{30}$$

and a fractional lead compensator corresponds to a D^{μ} controller with

$$K_d = K_L K_c, \quad \mu = \alpha_L \tag{31}$$

with the parameters ω_b and ω_h chosen according to the parameters b_L and a_L. It should be noted, that this method, as well as the choice of appropriate frequency bounds in (28), works only in case of the original Oustaloup filter in (7), not in case of the modified filter discussed in [10], [16].

C. Controller Reset Logic

Finally, we address the state reset logic for the IIR filters that implement the fractional-order controllers. Denote by $e(k)$ the kth sample of the error signal $e(\cdot)$. We propose the following filter memory reset logic based on the notion of a maximal error change rate margin ρ. The reset condition is expressed as follows

$$|e(k) - e(k-1)| > \rho. \tag{32}$$

Thus if the controller detects a sudden change in the error signal, IIR filter and integer-order integrator memory will be cleared, yielding zero initial conditions for the whole fractional-order PID controller or lead-lag compensator. It is important to select the value of the change rate margin ρ well above measurement noise or potential disturbance level.

VI. Hardware Implementation and Real-time Simulation Results

We use an Atmel AVR ATmega8A microcontroller to implement the results of this work. This choice is dictated by the relative popularity of this 8-bit microcontroller family and this microcontroller in particular. The chip is inexpensive and is widely used in industrial applications. We study the performance of this microcontroller for the fractional-order PID controller generation as well as DSP functions. Additionally, we employ external A/D and D/A converters in our FOPID controller prototype to bypass some DAQ hardware limitations of the microcontroller yielding a system with 12 bit sample resolution. The hardware configuration used in the real-time experiments is shown in Fig. 1.

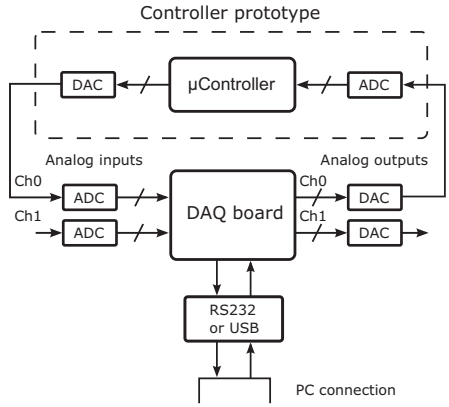


Fig. 1. Schematic diagram of the hardware used in real-time control experiments

The platform implements a hardware-in-the-loop scheme suited for verifying embedded hardware. Next, we provide a brief description of the platform:

- Simulink is used to model the plant under control. The Real-Time Windows Target toolbox is used to achieve real-time communication between the Simulink model and external controller prototype. This is done via the DAQ board controlled in real time from Simulink by means of a serial communication protocol. The maximum achievable sample rate is 10kSPS shared between 2 input and 2 output channels. Therefore, in case of a single input and a single output the sample rate is limited to 5kSPS in *Direct Port Access* mode. The current implementation allows for at most single-sample delays.
- The controller prototype is implemented on embedded hardware based on the ATmega8A microcontroller. The microcontroller is clocked at 16MHz and is running

firmware written in C-language and compiled with AVR-GCC with optimization level "O1". The prototype acts upon reception of analog signals from the DAQ board and generates a corresponding control law which is fed back to the Simulink model via the DAQ board. The input/output voltage signal range is $0 \dots 5$V.

In what follows, we provide performance evaluation figures based on the time it takes for the microcontroller to compute a particular FOPID approximation and to do a single sample computation.

A. Example of Fractional-order PID Controller Implementation

In this experiment, a model of a laboratory plant—coupled fluid tanks—is running in Simulink and it is our task to control this plant by means of our external controller prototype using a DAQ board. Details about the plant and earlier results are provided in [9].

We generate a FOPID with the following parameters:

$$K_p = 6.9514, \quad K_i = 0.13522, \quad K_d = -0.99874, \\ \lambda = 0.93187, \quad \mu = 0.29915. \quad (33)$$

The suitable frequency range for an Oustaloup filter of order $\nu = 5$ is $\omega = [0.0001, 10]\,\mathrm{rad/s}$ with $\nu_{max} = 10$. The sampling interval is $T_s = 0.2$s. Denote by τ_g and τ_s the time interval that is required for controller generation and sample computation, respectively, under the conditions above. We have the following per the report of AVR Simulator: $\tau_g = 27.6224\,\mathrm{ms}, \tau_s = 1.8904\,\mathrm{ms}$. Obviously, the controller must compute the next output sample faster than the specified sampling rate. Thus, sampling rates up to $f_s \approx 500$ Hz are possible in this case. Note, however, that it takes much longer to compute the coefficients of the controller. This should be considered when the controller is running in a closed loop and, consequently, in autotuning applications. In Table I a summary of time requirements for controller generation and sample computation for $\nu = 6, 7, \dots, 10$ is given.

TABLE I
TIME REQUIREMENTS FOR CONTROLLER GENERATION AND SAMPLE COMPUTATION FOR DIFFERENT OUSTALOUP FILTER ORDERS

ν	τ_g [ms]	τ_s [ms]	Max.applicable f_s [Hz]
6	32.2914	1.9868	480
7	37.1326	2.0832	450
8	42.2011	2.1796	425
9	46.8712	2.2759	400
10	51.5617	2.3723	400

Finally, the results of real-time controller evaluation are presented in Fig. 2. The graphs of two experiments are superimposed. The controller has been simulated and implemented on the external prototype. It can be seen, that the results are very close.

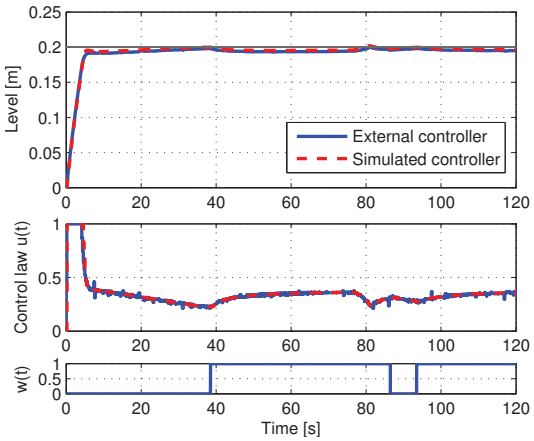


Fig. 2. Closed-loop control system response comparison: External FOPID vs. Simulated FOPID

B. Example of Fractional-order Lead Compensator Implementation

In this example, we implement a lead compensator using the method described in Sec. V-B. In the real-time experiment we consider a model of a position servo given in [10] which is represented by the following transfer function

$$G(s) = \frac{2}{s(0.5s+1)}. \quad (34)$$

A fractional lead compensator was designed in [10] such that

$$b_L = 0.6404, \quad a_L = 0.0032, \quad \alpha_L = 0.5 \quad (35)$$

with the design specifications crossover frequency $\omega_c = 10$rad/s and a phase margin of $\varphi_m = 50°$. An equivalent FOPID controller can be formulated such that

$$K_p = 0, \quad K_i = 0, \quad \lambda = 0, \\ K_d = 8.0025, \quad \mu = 0.5 \quad (36)$$

and an approximation is obtained with the following parameters:

$$\omega_b = 1.5615, \quad \omega_h = 312.50, \quad \nu = 5. \quad (37)$$

This approximation provides the following open-loop frequency-domain specifications of the control system: $\omega_c = 9.99$rad/s and $\varphi_m = 51°$. The corresponding Bode diagram is presented in Fig. 3.

The results of the real-time experiment with the digitally implemented fractional lead compensator are presented in Fig. 4. Once again, the graphs of two experiments, covering the evaluation of the hardware controller and the simulated one are superimposed. It can be seen, that the results are virtually identical.

VII. CONCLUSIONS

In this paper, we have discussed the important topic of digital approximations of fractional-order differential and integral operators. A discrete-time implementation method, based

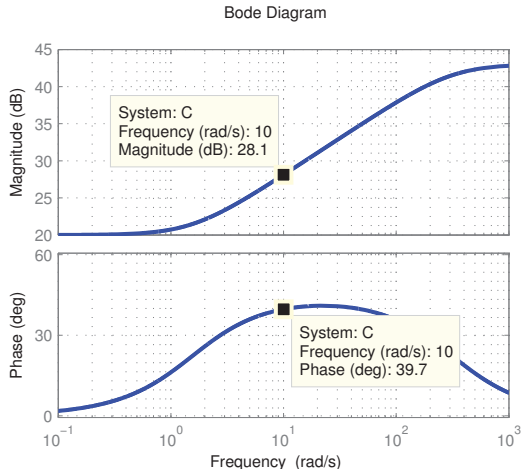


Fig. 3. Open-loop frequency-domain response of the obtained fractional lead compensator

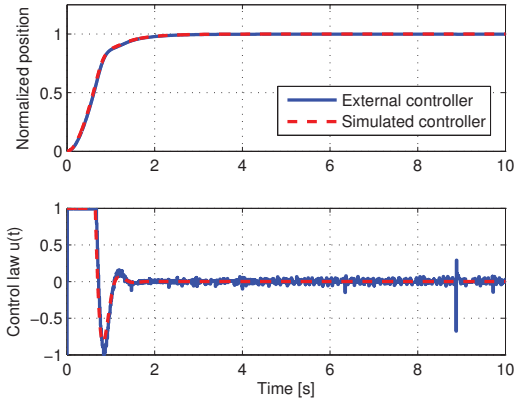


Fig. 4. Closed-loop control system response comparison: External FO lead compensator vs. Simulated FO lead compensator

on the Oustaloup recursive filter, was developed for embedded applications. A fractional-order PID controller generation scheme was also realized. Digital controller approximations of a fractional-order PID controller and a fractional-order lead-lag compensator were successfully verified on an Atmel AVR ATmega8A microcontroller. The proposed solution may be used in industrial control applications.

Since the controller and samples of the control law are computed using floating-point arithmetic, it is essential to carefully evaluate the methods involved in the computations to ensure numerical stability of the control algorithm. This is especially important when dealing with relatively low-end microprocessor systems.

Further research should cover automatic fractional-order PID controller tuning opportunities and implementation on other microcontroller families and other digital signal processing devices.

REFERENCES

[1] Y. Q. Chen, I. Petráš, and D. Xue, "Fractional order control - a tutorial," in *Proc. ACC '09. American Control Conference*, 2009, pp. 1397–1411.

[2] I. Podlubny, L. Dorčák, and I. Kostial, "On fractional derivatives, fractional-order dynamic systems and $PI^{\lambda}D^{\mu}$-controllers," in *Proc. 36th IEEE Conf. Decision and Control*, vol. 5, 1997, pp. 4985–4990.

[3] M. Čech and M. Schlegel, "The fractional-order PID controller outperforms the classical one," in *Process control 2006*. Pardubice Technical University, 2006, pp. 1–6.

[4] D. Xue, C. Zhao, and Y. Q. Chen, "Fractional order PID control of a DC-motor with elastic shaft: a case study," in *Proc. 2006 American Control Conference (ACC)*, 2006.

[5] Y. Luo and Y. Chen, "Fractional-order [proportional derivative] controller for robust motion control: Tuning procedure and validation," in *Proc. ACC '09. American Control Conference*, 2009, pp. 1412–1417.

[6] Y. Luo, Y. Q. Chen, H.-S. Ahn, and Y. Pi, "Fractional order robust control for cogging effect compensation in PMSM position servo systems: Stability analysis and experiments," *Control Engineering Practice*, vol. 18, no. 9, pp. 1022–1036, 2010.

[7] A. Oustaloup, F. Levron, B. Mathieu, and F. M. Nanot, "Frequency-band complex noninteger differentiator: characterization and synthesis," *IEEE Trans. Circuits Syst. I*, vol. 47, no. 1, pp. 25–39, 2000.

[8] A. Tepljakov, E. Petlenkov, and J. Belikov, "Implementation and real-time simulation of a fractional-order controller using a MATLAB based prototyping platform," in *Proc. 13th Biennial Baltic Electronics Conference*, 2012, pp. 145–148.

[9] A. Tepljakov, E. Petlenkov, J. Belikov, and M. Halás, "Design and implementation of fractional-order PID controllers for a fluid tank system," in *Proc. 2013 American Control Conference (ACC)*, Washington DC, USA, June 2013, pp. 1780–1785.

[10] C. A. Monje, Y. Q. Chen, B. M. Vinagre, D. Xue, and V. Feliu, *Fractional-order Systems and Controls: Fundamentals and Applications*, ser. Advances in Industrial Control. Springer Verlag, 2010.

[11] I. Petráš, S. Grega, and L. Dorčák, "Digital fractional order controllers realized by PIC microprocessor: Experimental results," in *Proc. of the ICCC'2003 conference*, High Tatras, Slovak Republic, 2003, pp. 873–876.

[12] A. Tepljakov, E. Petlenkov, and J. Belikov, "Application of Newton's method to analog and digital realization of fractional-order controllers," *International Journal of Microelectronics and Computer Science*, vol. 2, no. 2, pp. 45–52, 2012.

[13] A. Tepljakov, E. Petlenkov, J. Belikov, and J. Finajev, "Fractional-order controller design and digital implementation using FOMCON toolbox for MATLAB," in *Proc. of the 2013 IEEE Multi-Conference on Systems and Control conference*, 2013, pp. 340–345.

[14] A. Tepljakov, E. Petlenkov, and J. Belikov, "Tuning and digital implementation of a fractional-order PD controller for a position servo," *International Journal of Microelectronics and Computer Science*, vol. 4, no. 3, pp. 116–123, 2013.

[15] A. Tepljakov, E. Petlenkov, and J. Belikov. (2011) FOMCON toolbox. [Online]. Available: http://www.fomcon.net/

[16] D. Xue, Y. Q. Chen, and D. P. Atherton, *Linear Feedback Control: Analysis and Design with MATLAB (Advances in Design and Control)*, 1st ed. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2008.

[17] G. F. Franklin, M. L. Workman, and D. Powell, *Digital Control of Dynamic Systems*, 3rd ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1997.

Aleksei Tepljakov was born in 1987 in Tallinn. He received his B.Sc and M.Sc in Computer and Systems Engineering from Tallinn University of Technology in 2009 and 2011, respectively. He is currently a PhD student working at the Department of Computer Control at Tallinn University of Technology. His main research interests include fractional-order control of complex systems and fractional-order filter based analog and digital signal processing.

Eduard Petlenkov was born in 1979. He received his B.Sc, M.Sc and PhD degrees in computer and systems engineering from Tallinn University of Technology. He is an Associate Professor in the Department of Computer Control at Tallinn University of Technology. His main research interests lie in the domain of nonlinear control, system analysis and computational intelligence.

Juri Belikov was born in 1985. He received his B.Sc degree in mathematics from Tallinn University, and his M.Sc and PhD degrees in computer and systems engineering from Tallinn University of Technology. At present he holds the positions of researcher in the Institute of Cybernetics and Associate Professor in the Department at Computer Control of Tallinn University of Technology. His main research interests lie in the domain of nonlinear control theory.

Publication 6

Reference

A. Tepljakov, E. Petlenkov, J. Belikov, and E. A. Gonzalez, "Design of retuning fractional PID controllers for a closed-loop magnetic levitation control system," in ICARCV 2014: The 13th International Conference on Control, Automation, Robotics & Vision, 2014, pp. 1345–1350.

Abstract

In this paper, we study the problem of fractional-order PID controller design for an unstable plant—a laboratory model of a magnetic levitation system. To this end, we apply model based control design. A model of the magnetic levitation system is obtained by means of a closed-loop experiment. Several stable fractional-order controllers are identified and optimized by considering isolated stability regions. Finally, a nonintrusive controller retuning method is used to incorporate fractional-order dynamics into the existing control loop, thereby enhancing its performance. Experimental results confirm the effectiveness of the proposed approach. Control design methods offered in this paper are general enough to be applicable to a variety of control problems.

2014 13th International Conference on Control, Automation, Robotics & Vision
Marina Bay Sands, Singapore, 10-12th December 2014 (ICARCV 2014)

Design of Retuning Fractional PID Controllers for a Closed-loop Magnetic Levitation Control System

Aleksei Tepljakov and Eduard Petlenkov
Department of Computer Control
Tallinn University of Technology
Ehitajate tee 5, 19086 Tallinn, Estonia
E-mail: aleksei.tepljakov at ttu.ee
and eduard.petlenkov at ttu.ee

Juri Belikov
Institute of Cybernetics
Tallinn University of Technology
Akadeemia tee 21, 12618 Tallinn, Estonia
E-mail: jbelikov at cc.ioc.ee

Emmanuel A. Gonzalez
Jardine Schindler Elevator Corporation
8/F Pacific Star Bldg.,
Sen. Gil Puyat Ave. cor. Makati Ave.,
Makati City 1209 Philippines
E-mail: emm.gonzalez at delasalle.ph

***Abstract*—In this paper, we study the problem of fractional-order PID controller design for an unstable plant—a laboratory model of a magnetic levitation system. To this end, we apply model based control design. A model of the magnetic levitation system is obtained by means of a closed-loop experiment. Several stable fractional-order controllers are identified and optimized by considering isolated stability regions. Finally, a nonintrusive controller retuning method is used to incorporate fractional-order dynamics into the existing control loop, thereby enhancing its performance. Experimental results confirm the effectiveness of the proposed approach. Control design methods offered in this paper are general enough to be applicable to a variety of control problems.**

***Index Terms*—fractional-order calculus, PID control, unstable plant, stability analysis**

I. Introduction

Fractional-order calculus offers novel mathematical tools applicable to dynamical system modeling and control. This allows to achieve more accurate process models and more flexible controllers, thereby enhancing the quality of control loops. Since the majority of industrial control loops are of PI/PID type [1], it is of significant interest to study the problem of enhancing conventional PID controllers by introducing additional dynamical properties arising from making use of fractional-order integrators and differentiators. A controller of this type, called the fractional-order $\mathrm{PI}^{\lambda}\mathrm{D}^{\mu}$ controller (FOPID), was proposed by Podlubny in [2] and has since been a topic of active discussion in the control community [3]. Indeed, the additional freedom in tuning the controller allows to consider multiple robustness criteria. Therefore, a set of controller parameters can be obtained such, that fulfills several design specifications, which cannot be achieved by using a conventional PID controller [4]. More importantly, however, using fractional controllers grants the ability to obtain a wider set of stabilizing controller parameters, which is critical in case of unstable plants.

The Magnetic Levitation System (MLS) is a nonlinear, open-loop unstable, and time-varying system [5]. Therefore, designing a stabilizing controller for it is a challenging problem. Yet it is also of significant importance, since MLS has a considerable range of real-life applications—it is used in, e.g., high-speed magnetic levitation passenger trains and vibration isolation of sensitive machinery [6]. Corresponding nonlinear control design methods were proposed in, e.g., [7], [8], [9]. However, few research papers deal with control design for unstable systems [10], and, in particular, for the MLS, which forms the motivation for our present research effort.

We now summarize the contribution of this paper. First, a nonlinear model of the MLS is proposed, which is constructed based on several modeling approaches offered in literature [5], [10]. It is used in a model based control design method, which includes linear analysis around a working point, selecting random stabilizing FOPID controllers, heuristically detecting rectangular-shaped stability regions for pairs of controller gains, and obtaining suboptimal FOPID controller settings. The FOPID controller is then integrated into the control loop in a nonintrusive way, following the retuning method in [11]. Controller settings are verified on the real-life laboratory model of the MLS.

The paper is organized as follows. In Section II the reader is introduced to the mathematical tools of fractional-order calculus used throughout this paper. In Section III the nonlinear model of the MLS is presented. In Section IV the control design method, forming the main contribution of this paper, is described. Experimental results that verify the proposed control design approach follow in Section V. Finally, in Section VI conclusions are drawn.

II. Mathematical Tools

First, we consider fractional-order modeling. Fractional-order calculus is a generalization of integration and differentiation operations to the non-integer order operator ${}_a\mathscr{D}_t^{\alpha}$, where a and t are the lower and upper terminals of the operation, and α is the fractional order, such that

$$ {}_a\mathscr{D}_t^{\alpha} = \begin{cases} \frac{\mathrm{d}^{\alpha}}{\mathrm{d}t^{\alpha}} & \Re(\alpha) > 0, \\ 1 & \Re(\alpha) = 0, \\ \int_a^t (\mathrm{d}t)^{-\alpha} & \Re(\alpha) < 0, \end{cases} \tag{1} $$

where $\alpha \in \mathbb{R}^+$. The Laplace transform of $\mathscr{D}^{\alpha}$ of a signal $x(t)$ with zero initial conditions is given by

$$ \mathscr{L}\left\{\mathscr{D}^{\alpha} x(t)\right\} = s^{\alpha} X(s). \tag{2} $$

978-1-4799-5199-4/14/$31.00 ©2014 IEEE

A transfer function representation of a fractional dynamical model may be given by

$$G(s) = \frac{b_m s^{\beta_m} + b_{m-1} s^{\beta_{m-1}} + \cdots + b_0 s^{\beta_0}}{a_n s^{\alpha_n} + a_{n-1} s^{\alpha_{n-1}} + \cdots + a_0 s^{\alpha_0}}, \quad (3)$$

where usually $\beta_0 = \alpha_0 = 0$. The system in (3) has a commensurate order γ, such that $\lambda = s^\gamma$, if it can be represented in the following way:

$$H(\lambda) = \frac{\sum_{k=0}^{m} b_k \lambda^k}{\sum_{k=0}^{n} a_k \lambda^k}, \quad (4)$$

where n is called the pseudo-order of the system. The form (4) can also be used to determine the stability of the system by means of, e.g., Matignon's theorem [12], which is given next.

Theorem 1. *(Matignon's stability theorem) The fractional transfer function $G(s) = Z(s)/P(s)$ is stable if and only if the following condition is satisfied in σ-plane:*

$$|\arg(\sigma)| > q\frac{\pi}{2}, \forall \sigma \in C,\ P(\sigma) = 0, \quad (5)$$

where $\sigma := s^q$. When $\sigma = 0$ is a single root of $P(s)$, the system cannot be stable. For $q = 1$, this is the classical theorem of pole location in the complex plane: no pole is in the closed right plane of the first Riemann sheet.

It can be seen, that fractional-order systems offer a larger region of stability than conventional linear systems—roots of the characteristic polynomial $P(\sigma)$ may be located in the right half of the complex plane, as long as the condition (5) is satisfied. This theorem works for commensurate-order systems, where the commensurate order is given by q.

We now turn to fractional-order control. The parallel form of the PI$^\lambda$D$^\mu$ controller is given by

$$C(s) = K_p + K_i s^{-\lambda} + K_d s^\mu. \quad (6)$$

In this work, we consider the negative unity feedback closed loop system of the form

$$W(s) = \frac{C(s)G(s)}{1 + C(s)G(s)}, \quad (7)$$

where $C(s)$ is the PI$^\lambda$D$^\mu$ controller, and $G(s)$ is the plant under control.

Finally, in terms of implementation of fractional-order controllers we consider Oustaloup's approximation method, described in [13], which allows to obtain a band-limited approximation of a fractional-order operator in the form $s^\alpha \approx H(s)$, where $\alpha \in (-1, 1) \subset \mathbb{R}$ and $H(s)$ is a conventional linear, time-invariant system.

III. Model of the Magnetic Levitation System

The MLS consists of an electromagnet, a light source and sensor to measure the position of the levitated sphere, and a sphere rest, the height of which is variable and determines the initial position x_{max} of the sphere in control experiments.

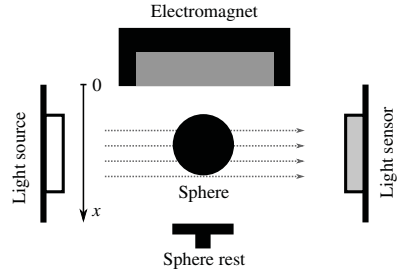


Fig. 1. Physical model of the MLS

The position of the sphere is deremined relative to the electromagnet and has an effective range of $x \in [0, x_{max}]$ mm. A schematic drawing depicting this configuration is given in Fig. 1. The basic principle of MLS operation is to apply voltage to the electromagnet to keep the sphere levitated [5].

In [6] and [10] the following dynamical model for the MLS is used:

$$m\ddot{x} = mg - \frac{ci^2(u)}{x^2}, \quad (8)$$

where m is the mass of the sphere, x is the position of the sphere, g is gravitational acceleration, $i(u)$ is a function of voltage corresponding to the electrical current running through the coil of the electromagnet under input u, and c is some constant. However, the following practical observations can be made:

- It is essential to model the dynamics of the electrical current running through the coil;
- The parameter c is, in fact, not constant and depends on the position of the sphere x.

Therefore, we use the model description provided by INTECO, which takes into account the dynamics of the coil current. In addition, we model the parameter c by a polynomial $c(x)$. The following model is finally established:

$$\begin{aligned} \dot{x}_1 &= x_2, \\ \dot{x}_2 &= -\frac{c(x_1)}{m}\frac{x_3^2}{x_1^2} + g, \\ \dot{x}_3 &= \frac{f_{ip2}}{f_{ip1}}\frac{i(u) - x_3}{\mathrm{e}^{-x_1/f_{ip2}}}, \end{aligned} \quad (9)$$

where x_1 is the position of the sphere, x_2 is the velocity of the sphere, and x_3 is the coil current, f_{ip1} and f_{ip2} are constants. By means of a series of experiments, we have found, that it is sufficient to model $c(x_1)$ as a 4th order polynomial of the form

$$c(x_1) = c_4 x_1^4 + c_3 x_1^3 + c_2 x_1^2 + c_1 x_1 + c_0, \quad (10)$$

and $i(u)$ as a 2nd order polynomial of the form

$$i(u) = k_2 u^2 + k_1 u + k_0. \quad (11)$$

Note, that the voltage control signal is normalized and has the range $u \in [0, 1]$ corresponding to the pulse-width modulation duty cycle $0 \ldots 100\%$.

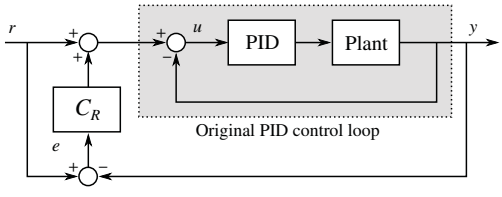


Fig. 2. Retuning method for existing closed-loop control systems

IV. Design and Implementation of Suboptimal Stabilizing Fractional-order PID Controllers

In the following, we summarize the method, that shall be used to design FOPID controllers for the MLS.

A. Model Linearization and Stability Analysis

We will analyze the stability of linear approximation around a working point (u_0, x_{10}). We linearize the model in (9) and obtain the following transfer function of the MLS:

$$G_M(s) = \frac{b_3 a_{23}}{s^3 - a_{33}s^2 - a_{21}s + a_{21}a_{33}}, \tag{12}$$

where

$$a_{21} = \frac{(-2c_4 x_{10}^4 - c_3 x_{10}^3 + c_1 x_{10} + 2c_0)x_{30}^2}{m x_{10}^3}, \tag{13}$$

$$a_{23} = -\frac{2c(x_{10})x_{30}}{m x_{10}^2}, \quad a_{33} = \frac{i(u_0) - x_{30}}{f_{ip1}} \mathrm{e}^{x_{10}/f_{ip2}}, \tag{14}$$

$$b_3 = \frac{f_{ip2}}{f_{ip1}}(k_1 + 2k_2 u_0)\mathrm{e}^{x_{10}/f_{ip2}}. \tag{15}$$

To analyze the stability of the closed-loop fractional-order control system in (7) we shall use Matignon's theorem. The characteristic polynomial is given by

$$\begin{aligned} Q(s) \;=\; & s^{3+\lambda} - a_{33}s^{2+\lambda} - a_{21}s \\ & + (b_3 a_{23} K_p + a_{21}a_{33})s^{\lambda} \\ & + b_3 a_{23} K_d s^{\lambda+\mu} + b_3 a_{23} K_i. \end{aligned} \tag{16}$$

Thus, a point of the form $(K_p, K_i, K_d, \lambda, \mu)$ in the PI$^\lambda$D$^\mu$ parameter space can be selected and the stability of the closed-loop control system can be verified.

B. PID Controller Retuning Method

The main idea of the retuning method is illustrated in Fig. 2. The method allows to incorporate fractional-order dynamics into a conventional PID control loop without making changes to the loop itself, but rather adding a second loop with the retuning FOPID controller. The following proposition establishes the relations between the parameters of the controllers [11].

Proposition 2. *Consider the original integer-order PID controller of the form*

$$C_{PID}(s) = K_P + K_I s^{-1} + K_D s. \tag{17}$$

Let $C_R(s)$ be a controller of the form

$$C_R(s) = \frac{K_2 s^\beta + K_1 s^\alpha - K_D s^2 + (K_0 - K_P)s - K_I}{K_D s^2 + K_P s + K_I}, \tag{18}$$

where the orders α and β are such, that $-1 < \alpha < 1$ and $1 < \beta < 2$. The PI$^\lambda$D$^\mu$ controller resulting from a classical PID controller will have the following coefficients

$$K_P^\star = K_0, \quad K_I^\star = K_1, \quad K_D^\star = K_2, \tag{19}$$

and the orders will be

$$\lambda = 1 - \alpha, \quad \mu = \beta - 1. \tag{20}$$

It can be shown, that the structure in Fig. 2 may be replaced by a feedback of the form (7), where

$$C(s) = (C_R(s) + 1) \cdot C_{PID}(s) \tag{21}$$

and $G(s)$ corresponds to the plant. Therefore, the parameters of the retuning controller $C_R(s)$ in (18) may be computed from those of the FOPID controller $C(s)$.

The application of the retuning method to the problem of control of the MLS is motivated by that we shall make use of closed-loop identification which may lead to a model that is sensitive to changes in parameters of the original PID controller. With the retuning method, a suitable controller is added into an external loop, and its control law is regulating the reference signal. Therefore, the underlying closed-loop system continues to operate as before, but the dynamics introduced to the reference signal allow to potentially enhance its performance.

C. Determination and Optimization of Stabilizing FOPID Controllers

To determine stabilizing controllers a randomized method may be used, where FOPID controller parameters are randomly selected from $K_p \in [K_p^l, K_p^u]$, $K_i \in [K_i^l, K_i^u]$, $K_d \in [K_d^l, K_d^u]$, $\lambda \in [\lambda^l, \lambda^u]$, $\mu \in [\mu^l, \mu^u]$. Note, that the choice of λ and μ must lead to a commensurate-order system, since only then the results of the stability test are reliable, otherwise they are only approximate [14]. For example, one can choose a minimum commensurate order $q = 0.01$.

Once a stable point is found, the following procedure is carried out. Two of the controller parameters are parametrized as (p_1, p_2), all other parameters are fixed. A limited number of steps N is selected and a sweep with step sizes Δp_1 and Δp_2 is done from the initial stable point. Four directions are considered. The main idea is illustrated in Fig. 3. Each time only a single parameter is changed. If, at any step, an unstable control loop is obtained, then the previous parameter value shall determine the approximate stability boundary for the corresponding direction. Otherwise, all points will be tested within the range $\Delta p_1 \cdot N$ and $\Delta p_2 \cdot N$. Testing is done by means of the characteristic polynomial in (16) and Matignon's theorem. Finally, the stability region will not always have a rectangular shape. Thus, it is possible to determine the shape by testing every point within the approximate rectangular stability boundary. This is a heuristic method similar to [15] and [16].

Once the procedure is complete, stable parameter ranges are obtained for all controller parameter pairs and may be used in FOPID controller optimization as lower and upper bounds

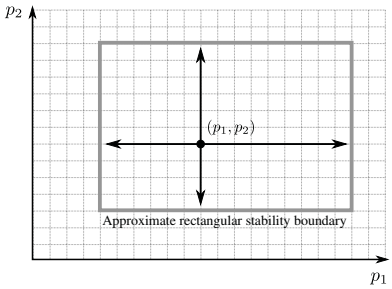


Fig. 3. Determination of the approximate rectangular stability boundary in the (p_1, p_2) plane

for corresponding controller parameters. Optimizing only two parameters at a time can be beneficial from the perspective of conditioning the problem, albeit in this case it will not be possible to satisfy several design constraints. Yet it poses great difficulty to impose feasible robustness specifications in case of the MLS. Thus, suboptimal controllers may be designed. The performance of the system will be evaluated experimentally, settling time τ_s, percent overshoot θ, and percent maximum deviation from reference due to disturbance θ_d are used as performance measures. In essence, we consider time-domain simulations of the nonlinear model in (9) and minimize a cost defined by

$$ISE = \int_0^t |e(\tau)| \mathrm{d}\tau, \tag{22}$$

where $e(\tau)$ is the error signal. The choice of this particular performance index is dictated by the necessity to minimize the overshoot [6]. The optimization procedure is carried out by means of the method described in [17], [18].

In what follows, we illustrate the proposed method on the basis of experimental results.

V. Experimental Results

For the purpose of validating our control design approach we use a real-life MLS provided by INTECO [5] and depicted in Fig. 4. It is connected to a computer running MATLAB/Simulink thereby allowing to conduct real-time experiments. The specific parameters of the model in (9) are as follows: $m = 0.0585$kg, $x_{max} = 0.0155$m, $g = 9.81$m/s^2. Other parameters need to be identified. The corresponding procedure is detailed in the following subsection.

A. Identification of the Nonlinear Model

Our task is to identify two functions $i(u)$ and $c(x)$, as well as parameters f_{ip1} and f_{ip2} of the nonlinear model in (9).

Identification of $i(u)$ is relatively simple and straightforward is done with the sphere removed from the MLS, since only the coil current is measured. We obtain the following polynomial:

$$i(u) = -0.3u^2 + 2.6u - 0.047. \tag{23}$$

In addition, the deadzone in control is found to be $u_{dz} = [0, 0.0182]$.

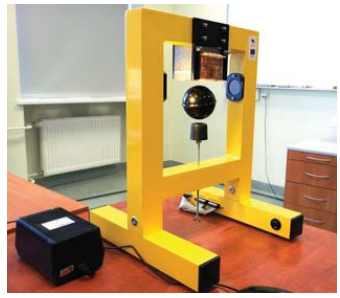


Fig. 4. Real-life laboratory model of the MLS

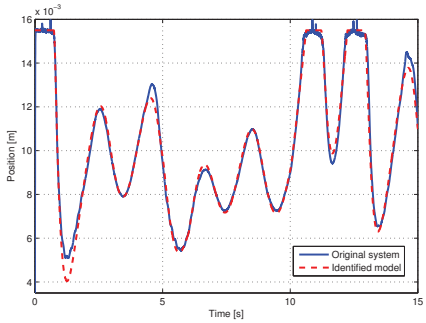


Fig. 5. Results of nonlinear model parameters identification

Determination of $c(x_1)$ and parameters f_{ip1} and f_{ip2}, on the other hand, is more involved. Because MLS is open-loop unstable, only closed-loop identification is applicable. Our approach is to use the existing PID control loop with

$$K_P = -39, \quad K_I = -10, \quad K_D = -2.05 \tag{24}$$

provided by INTECO. It should be noted, that a constant input $u_c = 0.38$ is added to the control law $u_{PID}(t)$ in (24), that is the full control law $u(t)$ is such that

$$u(t) = u_{PID}(t) + u_c. \tag{25}$$

In order to determine the values of the parameters, we employ time-domain simulations and minimize the model output error by means of the least-squares method. The results are as follows:

$$c(x_1) = 3.9996x_1^4 + 3.9248x_1^3 - 0.34183x_1^2 + 0.007058x_1 + 2.9682 \cdot 10^{-5} \tag{26}$$

and

$$f_{ip1} = 1.1165 \cdot 10^{-3}\text{m/s}, \quad f_{ip2} = 26.841 \cdot 10^{-3}\text{m}. \tag{27}$$

The results of the identification are presented in Fig. 5. It can be seen, that a close fit to the response of the original response of the system is achieved.

B. Design of FOPID Controllers

We first obtain a linear model as discussed in Section IV-A. We choose a working point $u_0 = 0.3726, x_{10} = 9.84 \cdot 10^{-3}$ and obtain

$$G_M(s) = -\frac{1788}{s^3 + 34.69s^2 - 1737s - 60240}. \quad (28)$$

Next, we apply the method detailed in Section IV-C. First, we randomly generate FOPID controllers using the ranges $K_p \in [-100, 0]$, $K_i \in [-50, 0]$, $K_d \in [-25, 0]$, $\lambda \in [0.8, 1.2]$, $\mu \in [0.5, 1.0]$. On the average, about 20 out of 100 tested controllers are found to produce a stable closed-loop system. After inspection, three of them are selected for the optimization phase:

$$C_1(s) = -42.8642 - \frac{18.5653}{s^{1.06}} - 3.0559s^{0.94}, \quad (29)$$

$$C_2(s) = -54.3649 - \frac{47.6078}{s^{0.82}} - 6.5436s^{0.98}, \quad (30)$$

$$C_3(s) = -45.3118 - \frac{4.24932}{s^{0.86}} - 3.51115s^{0.98}. \quad (31)$$

For each controller in this set, we find stability boundaries in different parameter planes, that is in (K_p, K_i), (K_p, K_d), and (K_i, K_d), so that we can obtain a wider set of results. Using the method in Section IV-C, with a step of $\Delta p = 1$ and considering a maximum of $N = 20$ steps we locate the following bounds:

$$K_p^{C_1} \in [-62, -34], \quad K_i^{C_1} \in [-38, -1], \quad (32)$$

$$K_p^{C_2} \in [-74, -35], \quad K_d^{C_2} \in [-26, -3], \quad (33)$$

$$K_i^{C_3} \in [-24, -1], \quad K_d^{C_3} \in [-23, -2]. \quad (34)$$

We then proceed directly to the optimization procedure. The FOMCON toolbox FOPID optimization tool is used [17], [19]. We set the bounds of controller gains as in (32)–(34) for each controller and optimize only the corresponding gains. The number of iterations is, in general, limited to $N_{iter} = 5$. After optimization, the following controllers are obtained:

$$C_1^*(s) = -45.839 - \frac{18.504}{s^{1.06}} - 3.0559s^{0.94}, \quad (35)$$

$$C_2^*(s) = -54.444 - \frac{47.6078}{s^{0.82}} - 3.7773s^{0.98}, \quad (36)$$

$$C_3^*(s) = -45.3118 - \frac{4.916}{s^{0.86}} - 2.9074s^{0.98}. \quad (37)$$

In the following, we provide the results of performance evaluation of both the randomly generated FOPID controllers, and the suboptimal ones. The controllers are evaluated in a two-cascade closed control loop as detailed in Section IV-B. The parameters of the retuning controllers are computed by means of (19) and (20). The performance of FOPID controllers is compared to the performance of the original PID control loop, where the parameters of the PID controller are equal to those in (24). The reference set point is $x_r = 0.010$m, and a disturbance impulse is considered, appearing for 200ms on the 10th second of the simulation. With the conventional PID controller the following results are achieved:

$$\tau_s = 3.34\,\text{s}, \quad \theta = 66.0\%, \quad \theta_d = 60.6\%.$$

Table I. Comparison of FOPID controller performance

FOPID	τ_s[s]	θ[%]	θ_d[%]	FOPID*	τ_s[s]	θ[%]	θ_d[%]
$C_1(s)$	1.85	24.0	60.3	$C_1^*(s)$	1.68	14.8	56.4
$C_2(s)$	**1.39**	**19.4**	**37.5**	$C_2^*(s)$	**0.86**	**11.6**	**34.6**
$C_3(s)$	4.68	14.6	55.7	$C_3^*(s)$	3.84	15.0	58.3

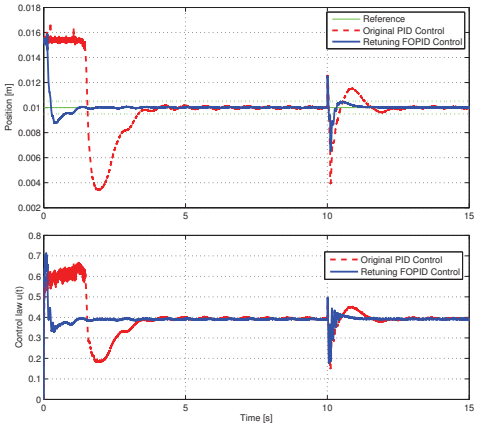


Fig. 6. Step experiment: Original PID control vs. Retuning FOPID controller

In Table I the performance evaluation of the FOPID controllers working in the retuning control loop is presented. It can be seen, that the best performance is achieved, when controller $C_2^*(s)$ is used. The result of real-time simulation of this controller versus the original PID control loop is provided in Fig. 6. It can be seen that a significant improvement in control system response is obtained. The controller $C_3(s)$ outperforms the original PID only in terms of overshoot, while $C_3^*(s)$ offers similar settling time with a much smaller overshoot.

In addition, we consider a reference tracking experiment to illustrate the ability of the controllers to provide appropriate regulation across a wider operating range. The comparison of the performance of the $C_2^*(s)$ controller and the original control loop is presented in Fig. 7. Once again, improvements in the control loop performance can be observed.

VI. Conclusions

In this paper, we have presented a method for FOPID controller design that allows incorporating fractional-order dynamics into existing PID control loops. An unstable plant, namely the MLS system was considered. A nonlinear model of this plant was identified from a closed-loop experiment. Linear analysis methods were employed to determine stabilizing

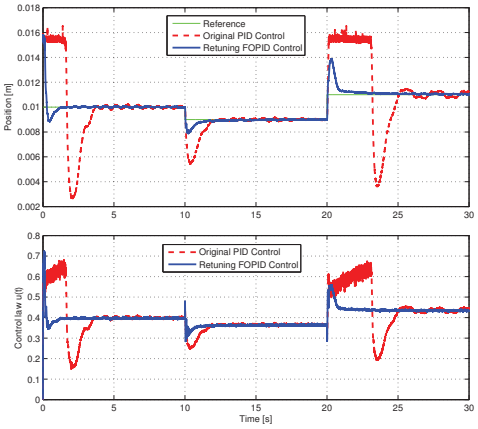


Fig. 7. Reference tracking: Original PID control vs. Retuning FOPID controller

FOPID controllers and stability boundaries in two-dimensional parameter planes thereof. The controllers were then evaluated, and those with best performance were optimized. In all cases, the optimization procedure enhanced the performance of the control loop. Virtually all retuning controllers offer superior performance compared to the original control loop, thereby establishing the validity of the proposed approach.

ACKNOWLEDGMENT

This work was partially supported by the Estonian Doctoral School in Information and Communication Technology.

REFERENCES

[1] K. Åström and T. Hägglund, *Advanced PID control*. The Instrumentation, Systems, and Automation Society (ISA), 2006.

[2] I. Podlubny, L. Dorčák, and I. Kostial, "On fractional derivatives, fractional-order dynamic systems and $PI^{\lambda}D^{\mu}$-controllers," in *Proc. 36th IEEE Conf. Decision and Control*, vol. 5, 1997, pp. 4985–4990.

[3] D. Xue and Y. Chen, "A comparative introduction of four fractional order controllers," in *Proceedings of the 4th World Congress on Intelligent Control and Automation*, 2002, pp. 3228–3235.

[4] C. A. Monje, Y. Q. Chen, B. M. Vinagre, D. Xue, and V. Feliu, *Fractional-order Systems and Controls: Fundamentals and Applications*, ser. Advances in Industrial Control. Springer Verlag, 2010.

[5] (2013) Official website of INTECO Sp. z o. o. [Last access time: 11.03.2013]. [Online]. Available: http://www.inteco.com.pl/

[6] H. Gole, P. Barve, A. A. Kesarkar, and N. Selvaganesan, "Investigation of fractional control performance for magnetic levitation experimental set-up," in *Emerging Trends in Science, Engineering and Technology (INCOSET), 2012 International Conference on*, Dec 2012, pp. 500–504.

[7] N. Al-Muthairi and M. Zribi, "Sliding mode control of a magnetic levitation system," *Mathematical Problems in Engineering*, vol. 2004, no. 2, pp. 93–107, 2004.

[8] E. Shameli, M. Khamesee, and J. Huissoon, "Nonlinear controller design for a magnetic levitation device," *Microsystem Technologies*, vol. 13, no. 8-10, pp. 831–835, 2007.

[9] V. Bandal and P. Vernekar, "Design of a discrete-time sliding mode controller for a magnetic levitation system using multirate output feedback," in *American Control Conference (ACC), 2010*, June 2010, pp. 4289–4294.

[10] C. I. Muresan, C. Ionescu, S. Folea, and R. Keyser, "Fractional order control of unstable processes: the magnetic levitation study case," *Nonlinear Dynamics*, pp. 1–12, 2014.

[11] E. A. Gonzalez, C. A. Monje, L. Dorčák, J. Terpák, and I. Petráš, "A method for incorporating fractional-order dynamics through pid control system retuning," *International Journal of Pure and Applied Mathematics*, vol. 68, no. 4, pp. 593–605, 2013.

[12] D. Matignon, "Generalized fractional differential and difference equations: Stability properties and modeling issues," in *Proc. of Math. Theory of Networks and Systems Symposium*, 1998, pp. 503–506.

[13] A. Oustaloup, F. Levron, B. Mathieu, and F. M. Nanot, "Frequency-band complex noninteger differentiator: characterization and synthesis," vol. 47, no. 1, pp. 25–39, 2000.

[14] Y. Q. Chen, I. Petráš, and D. Xue, "Fractional order control - a tutorial," in *Proc. ACC '09. American Control Conference*, 2009, pp. 1397–1411.

[15] Y. Lee and J. Watkins, "Determination of all stabilizing fractional-order PID controllers," in *American Control Conference (ACC), 2011*, 2011, pp. 5007–5012.

[16] Ü. Nurges, "New stability conditions via reflection coefficients of polynomials," *IEEE Transactions on Automatic Control*, vol. 50, no. 9, pp. 1354–1360, 2005.

[17] A. Tepljakov, E. Petlenkov, and J. Belikov, "A flexible MATLAB tool for optimal fractional-order PID controller design subject to specifications," in *Proceedings of the 31st Chinese Control Conference*, W. Li and Q. Zhao, Eds., Hefei, Anhui, China, 2012, pp. 4698–4703.

[18] A. Tepljakov, E. Petlenkov, J. Belikov, and M. Halás, "Design and implementation of fractional-order PID controllers for a fluid tank system," in *Proc. 2013 American Control Conference (ACC)*, Washington DC, USA, June 2013, pp. 1780–1785.

[19] A. Tepljakov, E. Petlenkov, and J. Belikov. (2011) FOMCON toolbox. [Online]. Available: http://www.fomcon.net/

Publication 7

Reference

A. Tepljakov, E. Petlenkov, and J. Belikov, "FOPID controlling tuning for fractional FOPDT plants subject to design specifications in the frequency domain," in Proc. 2015 European Control Conference (ECC), 2015, accepted for publication.

Abstract

In this paper, the problem of fractional-order proportional-integral-derivative (FOPID) controller tuning for fractional-order first-order plus dead time plants (FFOPDT) is considered. The designed controller must fulfill the prescribed specifications in the frequency domain. To this end, the fractional orders of the integrator and differentiator components are selected using a set of rules based on the observation of essential plant dynamics, while a numerical optimization algorithm is employed for obtaining the gains of the controller based on three dimensional Newton's method. All the necessary equations for computing the elements of the Jacobian matrix are provided. The proposed algorithm is detailed. It is also implemented and verified on an embedded device. The proposed solution may be useful in developing automatic tuning algorithms for FOPID controllers.

2015 European Control Conference (ECC)
July 15-17, 2015. Linz, Austria

FOPID Controller Tuning for Fractional FOPDT Plants subject to Design Specifications in the Frequency Domain

Aleksei Tepljakov[1], Eduard Petlenkov[1], and Juri Belikov[2]

***Abstract*— In this paper, the problem of fractional-order proportional-integral-derivative (FOPID) controller tuning for fractional-order first-order plus dead time plants (FFOPDT) is considered. The designed controller must fulfill the prescribed specifications in the frequency domain. To this end, the fractional orders of the integrator and differentiator components are selected using a set of rules based on the observation of essential plant dynamics, while a numerical optimization algorithm is employed for obtaining the gains of the controller based on three dimensional Newton's method. All the necessary equations for computing the elements of the Jacobian matrix are provided. The proposed algorithm is detailed. It is also implemented and verified on an embedded device. The proposed solution may be useful in developing automatic tuning algorithms for FOPID controllers.**

I. Introduction

PID-type controllers are widely used in industrial process control loops [1], [2]. However, it is a known fact that only a part of the existing PI/PID loops are adequately tuned ensuring the best possible performance [3]. Therefore, it is reasonable to invest proper attention to the tuning process by, e.g., applying an automatic tuning procedure to achieve improvement of control quality.

In process control it is common to use low-order models of the plant to be controlled, e.g., a first-order plus dead time (FOPDT) model, which captures essential process dynamics and allows one to design a suitable controller based on a set of particular tuning rules. However, it may not be sufficient, in general, to consider only classical dynamics, since conventional tuning rules may not lead to satisfactory control loop performance, or even fail. The recent emergence of fractional-order calculus has made it possible to make the transition from classical process models and PID controllers to those of noninteger order, allowing for better modeling and control design opportunities [4].

In terms of modeling, using fractional differential equations allows one to capture certain phenomena related to memory- and hereditary properties of the process [5]. Thus, a FFOPDT model was introduced and studied in, e.g., [6], [7]. Using fractional-order FOPID controllers introduced in [8] generally provides more tuning freedom. It has been confirmed that FOPID controllers offer superior performance than classical PID controllers [9], [10]. FOPID controller tuning has been discussed in, e.g., [11], [12], [13], [14]. Recently, FOPI controller design based on analysis of frequency-domain characteristics of the control system was proposed in [15]. In our earlier works we considered numerical optimization based FOPID controller design [16], [17]. In this paper we consider a particular class of fractional systems described by the FFOPDT model. We propose a particular FOPID controller design method based on exact control loop frequency-domain specifications.

In the following, we outline the main contribution of this paper. First, a set of rules for selecting the orders of the integrator and differentiator of the FOPID controller is proposed. Then, a system of three nonlinear equations in three unknowns is constructed. The solution of this system grants the gains of the FOPID controller. Since the system cannot be solved algebraically, the Newton method in multiple dimensions is adopted to the particular problem and a corresponding algorithm is outlined. The algorithm is verified on an embedded device. Finally, an example is provided, which illustrates the use of the method.

The paper is organized as follows. First, the reader is briefly introduced to the main concepts of fractional calculus regarding modeling and control in Section II. In Section III the necessary equations for control system analysis in the frequency domain are provided. The proposed controller design method is detailed in Section IV. An illustrative example follows in Section V. Items for discussion are provided in Section VI. Finally, conclusions are drawn in Section VII.

II. Introduction to Fractional-order Modeling and Control

Fractional calculus is a generalization of differential and integral operators to a noninteger order operator ${}_a\mathscr{D}_t^\alpha$, where α is the operator order and a, t denote operation limits [4]. The continuous integro-differential operator of order $\alpha \in \mathbb{R}$ is defined in the following way

$$ {}_a\mathscr{D}_t^\alpha = \begin{cases} \mathrm{d}^\alpha/\mathrm{d}t^\alpha & \alpha > 0, \\ 1 & \alpha = 0, \\ \int_a^t (\mathrm{d}\tau)^{-\alpha} & \alpha < 0. \end{cases} \tag{1} $$

Assuming zero initial conditions, the Laplace transform of the fractional derivative with $\alpha \in \mathbb{R}^+$ is given by

$$ \int_0^\infty \mathrm{e}^{-st}\, {}_0\mathscr{D}_t^\alpha f(t)\mathrm{d}t = s^\alpha F(s). \tag{2} $$

[1]Aleksei Tepljakov and Eduard Petlenkov are with the Department of Computer Control, Tallinn University of Technology, Tallinn, 19086, Estonia, {aleksei.tepljakov, eduard.petlenkov} at dcc.ttu.ee.

[2]Juri Belikov is with the Institute of Cybernetics, Tallinn University of Technology, Akadeemia tee 21, 12618, Tallinn, Estonia, jbelikov at cc.ioc.ee.

A fractional-order transfer function with a delay may be considered in the s-domain such that

$$G(s) = \frac{b_m s^{\beta_m} + b_{m-1} s^{\beta_{m-1}} + \cdots + b_0 s^{\beta_0}}{a_n s^{\alpha_n} + a_{n-1} s^{\alpha_{n-1}} + \cdots + a_0 s^{\alpha_0}} \mathrm{e}^{-Ls}, \quad (3)$$

where usually $\alpha_0 = \beta_0 = 0$, and in this case we denote by $K_g = b_0/a_0$ the static gain of the system. The frequency-domain characteristics of this transfer function may be obtained by substituting $s = j\omega$. We shall make use of this, additionally employing the relation

$$j^\gamma = \cos\left(\tfrac{\gamma\pi}{2}\right) + j \sin\left(\tfrac{\gamma\pi}{2}\right). \quad (4)$$

In this work we consider a particular version of the model (3), called by convention the FFOPDT model, with $m = 0$, $n = 1$, $\alpha_0 = \beta_0 = 0$, $K = b_0/a_0 > 0$, $T = a_1 > 0$, $L > 0$, and $\alpha = \alpha_1 \in (0, 2]$, and which is thereby given by the following transfer function

$$G_p(s) = \frac{K\mathrm{e}^{-Ls}}{Ts^\alpha + 1}. \quad (5)$$

Note, that with $\alpha = 1$ this reduces to the conventional FOPDT model. We shall maintain the notation for the parameters of this model throughout the paper.

The parallel form of the fractional-order PI$^\lambda$D$^\mu$ controller given is by

$$C(s) = K_p + K_i s^{-\lambda} + K_d \cdot s^\mu, \quad (6)$$

where $\lambda, \mu \in (0, 2]$. It can be seen, that by varying the non-integer order γ of a fractional-order operator s^γ a constant decrement or increment in the slope of the magnitude curve that equals 20γ dB/dec can be achieved, as well as a constant delay in the phase plot $\pi\gamma/2$ rad. In this paper, we study the usual negative unity feedback control system

$$G_c(s) = \frac{C(s)G_p(s)}{1 + C(s)G_p(s)}, \quad (7)$$

where $C(s)$ is the controller in (6) and $G_p(s)$ denotes the plant transfer function.

III. Frequency-domain Analysis of the Control Loop

Open-loop characteristics of the control system may be obtained by considering the magnitude and phase responses of the plant and controller. This is done by replacing $s = j\omega$, employing (4), and isolating the real and complex parts of the resulting expression as $z = a + jb$. The absolute value and the angle are then obtained in the usual manner.

We begin by providing the expressions for computation of the magnitude and phase responses of the plant $G(j\omega)$:

$$|G(j\omega)| = \frac{|K|}{\sqrt{1 + T^2\omega^{2\alpha} + 2T\omega^\alpha \cos\left(\frac{\alpha\pi}{2}\right)}} \quad (8)$$

and

$$\arg\left(G(j\omega)\right) = -L\omega - \tan^{-1}\left(\frac{T\sin\left(\frac{\alpha\pi}{2}\right)}{\omega^{-\alpha} + T\cos\left(\frac{\alpha\pi}{2}\right)}\right). \quad (9)$$

In the same way we derive the equations for the FOPID controller $C(j\omega)$ in (6):

$$|C(j\omega)| = \sqrt{C_R^2(\omega) + C_I^2(\omega)}, \quad (10)$$

where

$$C_R(\omega) = K_p + \omega^{-\lambda} K_i \cos\left(\tfrac{\lambda\pi}{2}\right) + \omega^\mu K_d \cos\left(\tfrac{\mu\pi}{2}\right) \quad (11)$$

and

$$C_I(\omega) = -\omega^{-\lambda} K_i \sin\left(\tfrac{\lambda\pi}{2}\right) + \omega^\mu K_d \sin\left(\tfrac{\mu\pi}{2}\right) \quad (12)$$

and for the phase angle as

$$\arg\left(C(j\omega)\right) = \tan^{-1}\left(\frac{C_N(\omega)}{C_D(\omega)}\right), \quad (13)$$

where

$$C_N(\omega) = \omega^{\lambda+\mu} K_d \sin\left(\tfrac{\mu\pi}{2}\right) - K_i \sin\left(\tfrac{\lambda\pi}{2}\right) \quad (14)$$

and

$$C_D(\omega) = K_i \cos\left(\tfrac{\lambda\pi}{2}\right) + \omega^\lambda \left(\omega^\mu K_d \cos\left(\tfrac{\mu\pi}{2}\right) + K_p\right). \quad (15)$$

Using the equations provided above it is possible to completely describe the frequency-domain characteristics of the control system. We now proceed to the proposed controller design method based on design specifications derived from these characteristics.

IV. Proposed Controller Design Method

We begin this section by outlining the setting. The parameters of the FFOPDT plant in (5) are assumed to be obtained by means of a relay autotuning algorithm considered in [12]. By properly identifying several points (ω_k, r_k) in the frequency domain it is possible to determine not only the classical FOPDT model, but also the fractional order α of the FFOPDT model. In case of the conventional model, where $\alpha = 1$ in (5), plant gain K_c and lag L_c may be determined experimentally [18] and will coincide with corresponding parameters of the FFOPDT model. The time constant T_c may be computed, when the frequency ω_g, corresponding to the ultimate gain of the system, is found, by means of

$$T_c = \frac{\tan(\pi - L_c\omega_g)}{\omega_g}. \quad (16)$$

Once all of the conventional FOPDT model parameters are obtained, we can use the F-MIGO rule proposed in [4] for FOPI controllers to determine the appropriate integrator order λ of the controller in (6) by considering basic plant dynamics through the relative dead time parameter τ_c:

$$\tau_c = \frac{L_c}{L_c + T_c}. \quad (17)$$

The following approximate rule is then employed:

$$\lambda = \begin{cases} 1.1, & \tau_c \geqslant 0.6, \\ 1.0, & 0.4 \leqslant \tau_c < 0.6, \\ 0.9, & 0.1 \leqslant \tau_c < 0.4, \\ 0.7, & \tau_c < 0.1. \end{cases} \quad (18)$$

This holds under the assumption that conventional plant dynamics are described by T_c with reasonable accuracy. Regardless of the situation, this is an approximation, therefore one may apply this rule to find an initial value of λ, which may be additionally tuned.

For deciding the differentiator order μ knowledge of the FFOPDT plant order α is necessary. The following relation should hold:

$$\mu \leqslant \alpha. \tag{19}$$

The particular value of μ may be chosen according to the design specifications imposed on the control system.

In this work, we consider the following specifications:

- Exact phase margin φ_m and corresponding crossover frequency ω_c, also referred to as critical frequency in [4];
- Robustness to gain variations, that is there exists a requirement such that

$$\psi_g'(\omega_c) = 0, \tag{20}$$

where

$$\psi_g(\omega) = \arg\left(C(j\omega)\right) + \arg\left(G(j\omega)\right) + \pi + 2\pi n, \tag{21}$$

the equations to compute (20) are provided in [19].
- Minimal gain margin G_m.

Based on these specifications, the following functions may be defined:

$$\kappa_1(K_p, K_i, K_d) = |C(j\omega)| \cdot |G(j\omega)| - 1, \tag{22}$$

$$\kappa_2(K_p, K_i, K_d) = \arg\left(C(j\omega)\right) + \arg\left(G(j\omega)\right) + \pi - \varphi_m - 2\pi n, \tag{23}$$

$$\kappa_3(K_p, K_i, K_d) = \psi_{gm}'(\omega), \tag{24}$$

where $\omega = \omega_c$. Note, that this does not include the gain margin specification. However, the solution is only considered feasible, if the minimal gain margin is satisfied.

To find the gains $g = \begin{bmatrix} K_p & K_i & K_d \end{bmatrix}^{\mathrm{T}}$ of the FOPID controller according to the specifications given above it is necessary to solve a system of nonlinear equations comprised of the design specification functions

$$F_s = \begin{bmatrix} \kappa_1(\cdot) & \kappa_2(\cdot) & \kappa_3(\cdot) \end{bmatrix}^{\mathrm{T}} = 0. \tag{25}$$

To this end, Newton's method in several dimensions may be employed. Beginning from the initial estimate g_0 the iterative process begins, until a particular stop condition is satisfied. On every step, a linear system

$$J\Delta g = -F_s \tag{26}$$

must be solved. Then the new controller gain vector g^+ is computed as

$$g^+ = g + \Delta g. \tag{27}$$

In the following, we provide all of the elements of the Jacobian matrix J comprised of the partial derivatives such that $J_{n,1} = \partial\kappa_n/\partial K_p$, $J_{n,2} = \partial\kappa_n/\partial K_i$, and $J_{n,3} = \partial\kappa_n/\partial K_d$, for $n = 1, 2, 3$:

$$J_{1,1} = \frac{A_G A_{CR}}{A_C},\ J_{1,2} = \frac{A_G A_{12}}{A_C},\ J_{1,3} = \frac{A_G A_{13}}{A_C}, \tag{28}$$

where $A_G = |G(j\omega)|$ in (8), $A_{CR} = C_R(\omega)$ in (11), and $A_C = |C(j\omega)|$ in (10),

$$A_{12} = \omega^{-2\lambda}\Big(-\omega^{\lambda+\mu}\sin\left(\tfrac{(\lambda+\mu-1)\pi}{2}\right)K_d + K_i + \omega^{\lambda}\cos\left(\tfrac{\lambda\pi}{2}\right)K_p\Big), \tag{29}$$

$$A_{13} = \omega^{-\lambda+\mu}\Big(\omega^{\lambda+\mu}K_d - \sin\left(\tfrac{(\lambda+\mu-1)\pi}{2}\right)K_i + \omega^{\lambda}\cos\left(\tfrac{\mu\pi}{2}\right)K_p\Big). \tag{30}$$

Then,

$$J_{2,1} = \frac{A_{21}}{A_2},\ J_{2,2} = -\frac{A_{22}}{A_2},\ J_{2,3} = \frac{A_{23}}{A_2}, \tag{31}$$

where

$$A_{21} = \omega^{\lambda}\left(-\omega^{\lambda+\mu}\sin\left(\tfrac{\mu\pi}{2}\right)K_d + \sin\left(\tfrac{\lambda\pi}{2}\right)K_i\right), \tag{32}$$

$$A_{22} = \omega^{\lambda}\Big(\omega^{\mu}\cos\left(\tfrac{(\lambda+\mu-1)\pi}{2}\right)K_d + \sin\left(\tfrac{\lambda\pi}{2}\right)K_p\Big), \tag{33}$$

$$A_{23} = \omega^{\lambda+\mu}\Big(\cos\left(\tfrac{(\lambda+\mu-1)\pi}{2}\right)K_i + \omega^{\lambda}\sin\left(\tfrac{\mu\pi}{2}\right)K_p\Big), \tag{34}$$

and

$$A_2 = \omega^{2(\lambda+\mu)}K_d^2 + K_i^2 + 2\omega^{\lambda}\cos\left(\tfrac{\lambda\pi}{2}\right)K_iK_p + \omega^{2\lambda}K_p^2 + 2\omega^{\lambda+\mu}K_d\Big(-\sin\left(\tfrac{(\lambda+\mu-1)\pi}{2}\right)K_i + \omega^{\lambda}\cos\left(\tfrac{\mu\pi}{2}\right)K_p\Big). \tag{35}$$

Finally,

$$J_{3,1} = \frac{A_{31}}{A_3},\ J_{3,2} = \frac{A_{32}}{A_3},\ J_{3,3} = \frac{A_{33}}{A_3}, \tag{36}$$

where

$$\begin{aligned} A_{31} = {} & \omega^{\lambda-1}\Big(\mu\omega^{3(\lambda+\mu)}\sin\left(\tfrac{\mu\pi}{2}\right)K_d^3 \\ & - \omega^{2(\lambda+\mu)}\Big(2\mu\sin\left(\tfrac{\lambda\pi}{2}\right) + \lambda\sin\left(\tfrac{(\lambda+2\mu)\pi}{2}\right)\Big)K_d^2K_i \\ & + \lambda\sin\left(\tfrac{\lambda\pi}{2}\right)K_i\Big(K_i^2 - \omega^{2\lambda}K_p^2\Big) \\ & - \omega^{\lambda+\mu}K_d\Big(\Big(2\lambda\sin\left(\tfrac{\mu\pi}{2}\right) + \mu\sin\left(\tfrac{(2\lambda+\mu)\pi}{2}\right)\Big)K_i^2 \\ & + 2(\lambda+\mu)\omega^{\lambda}\cos\left(\tfrac{(\lambda+\mu-1)\pi}{2}\right)K_iK_p \\ & + \mu\omega^{2\lambda}\sin\left(\tfrac{\mu\pi}{2}\right)K_p^2\Big)\Big), \end{aligned} \tag{37}$$

$$
\begin{aligned}
A_{32} = {} & \omega^{\lambda-1}\Big((\lambda+\mu)\omega^{2\lambda+3\mu}\cos\Big(\tfrac{(\lambda+\mu-1)\pi}{2}\Big)K_d^3 \\
& + \omega^{2(\lambda+\mu)}\Big(2(\lambda+\mu)\sin\big(\tfrac{\lambda\pi}{2}\big) \\
& + \lambda\sin\Big(\tfrac{(\lambda+2\mu)\pi}{2}\Big)\Big)K_d^2K_p \\
& + \lambda\sin\big(\tfrac{\lambda\pi}{2}\big)K_p(-K_i^2+\omega^{2\lambda}K_p^2) \\
& + \omega^{\mu}K_d\Big(-(\lambda+\mu)\cos\Big(\tfrac{(\lambda+\mu-1)\pi}{2}\Big)K_i^2 \\
& - 2\mu\omega^{\lambda}\sin\big(\tfrac{\mu\pi}{2}\big)K_iK_p \\
& + \omega^{2\lambda}\Big(2\lambda\cos\Big(\tfrac{(\lambda+\mu-1)\pi}{2}\Big) \\
& + (\lambda+\mu)\sin\Big(\tfrac{(\lambda-\mu)\pi}{2}\Big)\Big)K_p^2\Big)\Big), \qquad (38)
\end{aligned}
$$

$$
\begin{aligned}
A_{33} = {} & \omega^{\lambda+\mu-1}\Big((\lambda+\mu)\cos\Big(\tfrac{(\lambda+\mu-1)\pi}{2}\Big)K_i^3 \\
& - 2\lambda\omega^{2\lambda+\mu}\sin\big(\tfrac{\lambda\pi}{2}\big)K_pK_iK_d \\
& + \omega^{\lambda}\Big(2(\lambda+\mu)\sin\big(\tfrac{\mu\pi}{2}\big)+\mu\sin\Big(\tfrac{(2\lambda+\mu)\pi}{2}\Big)\Big)K_i^2K_p \\
& + \omega^{2\lambda}\Big(2\mu\cos\Big(\tfrac{(\lambda+\mu-1)\pi}{2}\Big) \\
& - (\lambda+\mu)\sin\Big(\tfrac{(\lambda-\mu)\pi}{2}\Big)\Big)K_iK_p^2 + \mu\omega^{3\lambda}\sin\big(\tfrac{\mu\pi}{2}\big)K_p^3 \\
& - \omega^{2(\lambda+\mu)}K_d^2\Big((\lambda+\mu)\cos\Big(\tfrac{(\lambda+\mu-1)\pi}{2}\Big)K_i \\
& + \mu\omega^{\lambda}\sin\big(\tfrac{\mu\pi}{2}\big)K_p\Big)\Big), \qquad (39)
\end{aligned}
$$

and

$$
A_3 = A_2^2. \qquad (40)
$$

We can define a stopping criterion for the iterative process as a condition on the square norm of (25):

$$
\|F_s(\cdot)\|_2 < \epsilon. \qquad (41)
$$

There is no feasible way to determine whether a solution to (25) exists in general or not. However, by virtue of the Inverse Value Theorem it is possible to claim, that if the Jacobian does not vanish at g_0, a local minimum of $\|F_s(\cdot)\|_2$ will be found around g_0. It is up to the user to check, whether the obtained set of controller parameters is feasible or not. If no feasible solution is obtained during optimization, a new initial estimate should be selected.

The complete optimization algorithm is presented in Fig. 1. The meaning of procedure return codes is provided in Table I.

procedure FOPIDDESIGN($g_0, \omega_c, \varphi_m, G_m$)
  $\epsilon \leftarrow$ Tolerance, $\epsilon_m \leftarrow$ MachineTolerance
  $g \leftarrow g_0, k \leftarrow 0, \nu \leftarrow$ MaxIterations
  while $k < \nu$ **do**
    if $\det J < \epsilon_m$ **then return** $\{-1, g\}$
    end if
    if $G_m^* < G_m$ **then return** $\{-2, g\}$
    end if
    if $\|F_s\|_2 < \epsilon$ **then return** $\{1, g\}$
    end if
    $g \leftarrow g - J^{-1}F_s$
    $k \leftarrow k + 1$
  end while
  return $\{0, g\}$
end procedure

Fig. 1. Determination of FOPID controller gains

TABLE I
MEANING OF OPTIMIZATION PROCEDURE RETURN CODES

Code	Description
-2	Additional condition not satisfied—the gain margin G_m^* computed for the control system is less than the value given in G_m.
-1	Singular Jacobian matrix—local minimum possible.
0	Maximum number of algorithm iterations reached.
1	All conditions satisfied, successful termination.

V. ILLUSTRATIVE EXAMPLE

As an example we consider a heating process described in [6]. It is given by the following FFOPDT transfer function:

$$
G(s) = \frac{66.16\mathrm{e}^{-1.93s}}{12.72s^{0.5}+1}. \qquad (42)
$$

The design specifications are as follows: $\omega_c = 0.1$, $\varphi_m = 60^\circ$, $G_m \geqslant 10\mathrm{dB}$, and the robustness to gain variations criterion must also be fulfilled.

Suppose that an autotuning procedure is employed. For the plant (42) we found the ultimate frequency ω_g to be approximately equal to 7.85rad/s. Using (16) the time constant of the conventional FOPDT plant would be computed as $T_c = 0.0794$s, which obviously provides the wrong description of the plant dynamics. Therefore, using the F-MIGO rule provided in Section IV does not hold any merit. Using a more sophisticated approach for computing T_c involving identification of several points on the Nyquist curve, however, yields a value $\tilde{T}_c \approx 12$, therefore we have $\lambda = 0.9$. The differentiator order is then chosen as $\mu = 0.5$. The initial gains for optimization are selected such that $g_0 = \begin{bmatrix} 1/K & 1/K & 1/K \end{bmatrix}$. The optimization procedure is then employed. In 4 iterations the norm condition $\epsilon = 10^{-4}$ is satisfied, and the gains of the resulting controller are obtained:

$$
K_p = -0.002934,\ K_i = 0.01030,\ K_d = 0.05335. \qquad (43)
$$

The Bode diagram depicting the open-loop frequency response of the control system is given in Fig. 3. It may be seen, that the specifications are fulfilled, with $G_m = 11.5 > 10\mathrm{dB}$.

The described algorithm was also implemented on an Atmel ATmega324P microcontroller based FOPID controller prototype device. In the following, some initial benchmarking results are provided. The C code is compiled using AVR-GCC with optimization option "-O1". In the example

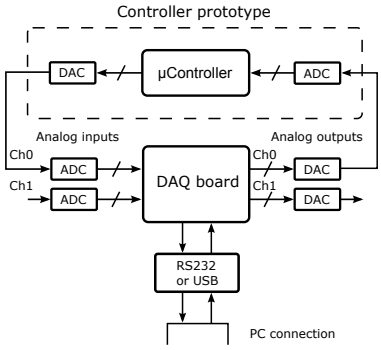


Fig. 2. Prototyping platform: connection diagram

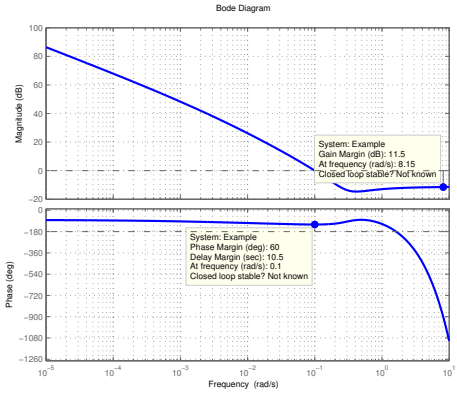


Fig. 3. Controller design results: Bode diagram of the open-loop control system

problem above, the very same gains were obtained in 4 algorithm iterations in approximately 1.66M clock cycles, or 83.37ms with the microcontroller clocked at 20MHz.

The resulting controller was also verified by means of pure software and hardware-in-the-loop (HIL) real-time simulations. Pure software simulations are done in MATLAB/Simulink environment. For the HIL part, a prototyping platform was used which is depicted in Fig. 2. In this configuration, the process model (42) is running in MATLAB/Simulink environment, while the controller prototype is connected to it externally via a DAQ board.

The results of real-time simulations are shown in Figs. 4 and 5. The gain of the system in (42) is varying in the range $\pm 25\%$. The following observations can be made. First, as expected, the overshoot value does not change significantly between experiments, where the plant gain is different. This shows, that the iso-damping property in (20) is indeed satisfied. Second, the corresponding results of pure software and HIL simulations are very close, which points to the adequate implementation of the controller prototype.

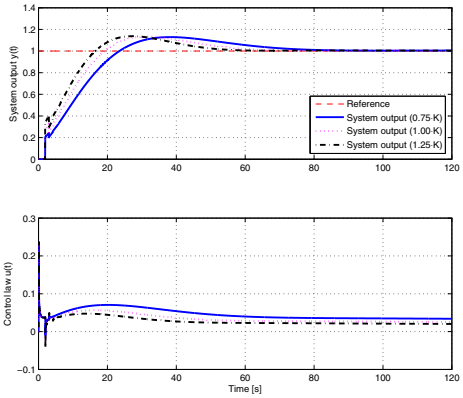


Fig. 4. Pure software simulations of the FOPID control system with varying gain ($\tilde{K} = \{0.75K, 1.00K, 1.25K\}$)

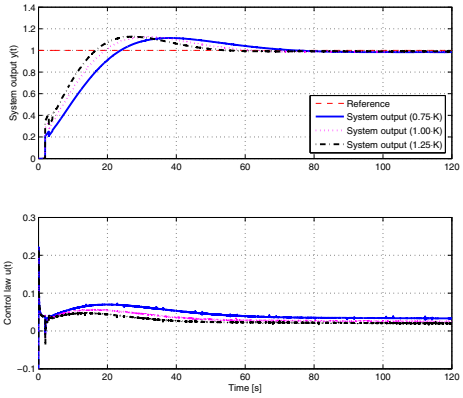


Fig. 5. Hardware-in-the-loop simulations of the FOPID control system with varying gain ($\tilde{K} = \{0.75K, 1.00K, 1.25K\}$)

VI. Discussion

In the following, several items for discussion are given. These mostly concern the improvement of selection rules for FOPID controller parameters λ and μ.

- The use of F-MIGO rule needs to be tailored to the case of the FFOPDT model. Currently this rule can only provide approximate results, since a conventional FOPDT model is used to obtain the parameter λ;
- A more sophisticated method for choosing the order of the differentiator μ should also be sought.

It is possible to consider more design specifications in the tuning process described in this paper. However, this makes it necessary to include parameters λ and μ as unknowns into the nonlinear equations. Partial derivatives involving

these parameters are nontrivial and tedious. Our current research shows, that employing numerical approximations of the Jacobian does not yield satisfactory results with this method.

VII. Conclusion

In this paper, we have presented a method for designing FOPID controllers for FFOPDT plants based on available autotuning data. The tuning method was validated on a model of a heating process and successfully implemented on an embedded device. Experimental results involving real-time hardware-in-the-loop simulations confirm the validity of the proposed approach.

A problem with conventional tuning approach was found, where the time constant of the system was not correctly identified. A more sophisticated tuning approach, which falls outside of the scope of this paper, may be used to tackle the issue. This problem forms an important part of our further research of FOPID controller autotuning methods.

In addition, research efforts should also be dedicated to developing a set of rules for selecting the orders of the FOPID controller integrator and differentiator components.

Acknowledgments

This work was partially supported by the Estonian Doctoral School in Information and Communication Technology through the interdisciplinary project FOMCON.

References

[1] K. Åström and T. Hägglund, *PID Controllers: Theory, Design, and Tuning*, 2nd ed. The Instrumentation, Systems, and Automation Society (ISA), 1995.

[2] V. Vansovits, E. Petlenkov, K. Vassiljeva, and A. Guljajev, "Identification of industrial water boiler for model predictive control of district heat plant," in *Proceedings of the 13th biennial Baltic Electronics Conference*, 2012, pp. 314–318.

[3] K. Åström and T. Hägglund, *Advanced PID control*. The Instrumentation, Systems, and Automation Society (ISA), 2006.

[4] C. A. Monje, Y. Q. Chen, B. M. Vinagre, D. Xue, and V. Feliu, *Fractional-order Systems and Controls: Fundamentals and Applications*, ser. Advances in Industrial Control. Springer Verlag, 2010.

[5] Y. Q. Chen, I. Petráš, and D. Xue, "Fractional order control - a tutorial," in *Proc. ACC '09. American Control Conference*, 2009, pp. 1397–1411.

[6] H. Malek, Y. Luo, and Y. Q. Chen, "Tuning fractional order proportional integral controllers for time delayed systems with a fractional pole," in *Proceedings of the ASME 2011 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, vol. 3, 2011, pp. 1–11.

[7] F. Padula, R. Vilanova, and A. Visioli, "H_∞ model matching PID design for fractional FOPDT systems," in *American Control Conference (ACC), 2012*, 2012, pp. 5513–5518.

[8] I. Podlubny, L. Dorčák, and I. Kostial, "On fractional derivatives, fractional-order dynamic systems and $PI^\lambda D^\mu$-controllers," in *Proc. 36th IEEE Conf. Decision and Control*, vol. 5, 1997, pp. 4985–4990.

[9] D. Xue, C. Zhao, and Y. Q. Chen, "Fractional order PID control of a DC-motor with elastic shaft: a case study," in *Proc. 2006 American Control Conference (ACC)*, 2006.

[10] M. Čech and M. Schlegel, "The fractional-order PID controller outperforms the classical one," in *Process control 2006*. Pardubice Technical University, 2006, pp. 1–6.

[11] C. Zhao, D. Xue, and Y. Q. Chen, "A fractional order PID tuning algorithm for a class of fractional order plants," in *Proc. IEEE Int. Mechatronics and Automation Conf.*, vol. 1, 2005, pp. 216–221.

[12] C. A. Monje, B. M. Vinagre, V. Feliu, and Y. Q. Chen, "Tuning and auto-tuning of fractional order controllers for industry applications," *Control Engineering Practice*, vol. 16, no. 7, pp. 798–812, 2008.

[13] R. Caponetto, G. Dongola, L. Fortuna, and A. Gallo, "New results on the synthesis of FO-PID controllers," *Communications in Nonlinear Science and Numerical Simulation*, vol. 15, no. 4, pp. 997–1007, 2010.

[14] Y. Luo and Y. Chen, "Stabilizing and robust fractional order PI controller synthesis for first order plus time delay systems," *Automatica*, vol. 48, no. 9, pp. 2159–2167, 2012.

[15] V. Feliu-Batlle and F. J. Castillo-García, "On the robust control of stable minimum phase plants with large uncertainty in a time constant. A fractional-order control approach," *Automatica*, vol. 50, no. 1, pp. 218–224, 2014.

[16] A. Tepljakov, E. Petlenkov, and J. Belikov, "FOMCON: Fractional-order modeling and control toolbox for MATLAB," in *Proc. 18th Int. Mixed Design of Integrated Circuits and Systems (MIXDES) Conference*, A. Napieralski, Ed., 2011, pp. 684–689.

[17] A. Tepljakov, E. Petlenkov, J. Belikov, and M. Halás, "Design and implementation of fractional-order PID controllers for a fluid tank system," in *Proc. 2013 American Control Conference (ACC)*, Washington DC, USA, June 2013, pp. 1780–1785.

[18] C. Yu, *Autotuning of PID Controllers: A Relay Feedback Approach*. Springer, 2006.

[19] A. Tepljakov, E. Petlenkov, and J. Belikov, "Robust FOPI and FOPID controller design for FFOPDT plants in embedded control applications using frequency-domain analysis," in *Proc. 2015 American Control Conference (ACC)*, 2015, accepted for publication.

DISSERTATIONS DEFENDED AT TALLINN UNIVERSITY OF TECHNOLOGY ON *INFORMATICS AND SYSTEM ENGINEERING*

1. **Lea Elmik**. Informational Modelling of a Communication Office. 1992.

2. **Kalle Tammemäe**. Control Intensive Digital System Synthesis. 1997.

3. **Eerik Lossmann**. Complex Signal Classification Algorithms, Based on the Third-Order Statistical Models. 1999.

4. **Kaido Kikkas**. Using the Internet in Rehabilitation of People with Mobility Impairments – Case Studies and Views from Estonia. 1999.

5. **Nazmun Nahar**. Global Electronic Commerce Process: Business-to-Business. 1999.

6. **Jevgeni Riipulk**. Microwave Radiometry for Medical Applications. 2000.

7. **Alar Kuusik**. Compact Smart Home Systems: Design and Verification of Cost Effective Hardware Solutions. 2001.

8. **Jaan Raik**. Hierarchical Test Generation for Digital Circuits Represented by Decision Diagrams. 2001.

9. **Andri Riid**. Transparent Fuzzy Systems: Model and Control. 2002.

10. **Marina Brik**. Investigation and Development of Test Generation Methods for Control Part of Digital Systems. 2002.

11. **Raul Land**. Synchronous Approximation and Processing of Sampled Data Signals. 2002.

12. **Ants Ronk**. An Extended Block-Adaptive Fourier Analyser for Analysis and Reproduction of Periodic Components of Band-Limited Discrete-Time Signals. 2002.

13. **Toivo Paavle**. System Level Modeling of the Phase Locked Loops: Behavioral Analysis and Parameterization. 2003.

14. **Irina Astrova**. On Integration of Object-Oriented Applications with Relational Databases. 2003.

15. **Kuldar Taveter**. A Multi-Perspective Methodology for Agent-Oriented Business Modelling and Simulation. 2004.

16. **Taivo Kangilaski**. Eesti Energia käiduhaldussüsteem. 2004.

17. **Artur Jutman**. Selected Issues of Modeling, Verification and Testing of Digital Systems. 2004.

18. **Ander Tenno**. Simulation and Estimation of Electro-Chemical Processes in Maintenance-Free Batteries with Fixed Electrolyte. 2004.

19. **Oleg Korolkov**. Formation of Diffusion Welded Al Contacts to Semiconductor Silicon. 2004.

20. **Risto Vaarandi**. Tools and Techniques for Event Log Analysis. 2005.

21. **Marko Koort**. Transmitter Power Control in Wireless Communication Systems. 2005.

22. **Raul Savimaa**. Modelling Emergent Behaviour of Organizations. Time-Aware, UML and Agent Based Approach. 2005.

23. **Raido Kurel**. Investigation of Electrical Characteristics of SiC Based Complementary JBS Structures. 2005.

24. **Rainer Taniloo**. Ökonoomsete negatiivse diferentsiaaltakistusega astmete ja elementide disainimine ja optimeerimine. 2005.

25. **Pauli Lallo**. Adaptive Secure Data Transmission Method for OSI Level I. 2005.

26. **Deniss Kumlander**. Some Practical Algorithms to Solve the Maximum Clique Problem. 2005.

27. **Tarmo Veskioja**. Stable Marriage Problem and College Admission. 2005.

28. **Elena Fomina**. Low Power Finite State Machine Synthesis. 2005.

29. **Eero Ivask**. Digital Test in WEB-Based Environment 2006.

30. **Виктор Войтович**. Разработка технологий выращивания из жидкой фазы эпитаксиальных структур арсенида галлия с высоковольтным p-n переходом и изготовления диодов на их основе. 2006.

31. **Tanel Alumäe**. Methods for Estonian Large Vocabulary Speech Recognition. 2006.

32. **Erki Eessaar**. Relational and Object-Relational Database Management Systems as Platforms for Managing Softwareengineering Artefacts. 2006.

33. **Rauno Gordon**. Modelling of Cardiac Dynamics and Intracardiac Bio-impedance. 2007.

34. **Madis Listak**. A Task-Oriented Design of a Biologically Inspired Underwater Robot. 2007.

35. **Elmet Orasson**. Hybrid Built-in Self-Test. Methods and Tools for Analysis and Optimization of BIST. 2007.

36. **Eduard Petlenkov**. Neural Networks Based Identification and Control of Nonlinear Systems: ANARX Model Based Approach. 2007.

37. **Toomas Kirt**. Concept Formation in Exploratory Data Analysis: Case Studies of Linguistic and Banking Data. 2007.

38. **Juhan-Peep Ernits**. Two State Space Reduction Techniques for Explicit State Model Checking. 2007.

39. **Innar Liiv**. Pattern Discovery Using Seriation and Matrix Reordering: A Unified View, Extensions and an Application to Inventory Management. 2008.

40. **Andrei Pokatilov**. Development of National Standard for Voltage Unit Based on Solid-State References. 2008.

41. **Karin Lindroos**. Mapping Social Structures by Formal Non-Linear Information Processing Methods: Case Studies of Estonian Islands Environments. 2008.

42. **Maksim Jenihhin**. Simulation-Based Hardware Verification with High-Level Decision Diagrams. 2008.

43. **Ando Saabas**. Logics for Low-Level Code and Proof-Preserving Program Transformations. 2008.

44. **Ilja Tšahhirov**. Security Protocols Analysis in the Computational Model – Dependency Flow Graphs-Based Approach. 2008.

45. **Toomas Ruuben**. Wideband Digital Beamforming in Sonar Systems. 2009.

46. **Sergei Devadze**. Fault Simulation of Digital Systems. 2009.

47. **Andrei Krivošei**. Model Based Method for Adaptive Decomposition of the Thoracic Bio-Impedance Variations into Cardiac and Respiratory Components. 2009.

48. **Vineeth Govind**. DfT-Based External Test and Diagnosis of Mesh-like Networks on Chips. 2009.

49. **Andres Kull**. Model-Based Testing of Reactive Systems. 2009.

50. **Ants Torim**. Formal Concepts in the Theory of Monotone Systems. 2009.

51. **Erika Matsak**. Discovering Logical Constructs from Estonian Children Language. 2009.

52. **Paul Annus**. Multichannel Bioimpedance Spectroscopy: Instrumentation Methods and Design Principles. 2009.

53. **Maris Tõnso**. Computer Algebra Tools for Modelling, Analysis and Synthesis for Nonlinear Control Systems. 2010.

54. **Aivo Jürgenson**. Efficient Semantics of Parallel and Serial Models of Attack Trees. 2010.

55. **Erkki Joasoon**. The Tactile Feedback Device for Multi-Touch User Interfaces. 2010.

56. **Jürgo-Sören Preden**. Enhancing Situation – Awareness Cognition and Reasoning of Ad-Hoc Network Agents. 2010.

57. **Pavel Grigorenko**. Higher-Order Attribute Semantics of Flat Languages. 2010.

58. **Anna Rannaste**. Hierarcical Test Pattern Generation and Untestability Identification Techniques for Synchronous Sequential Circuits. 2010.

59. **Sergei Strik**. Battery Charging and Full-Featured Battery Charger Integrated Circuit for Portable Applications. 2011.

60. **Rain Ottis**. A Systematic Approach to Offensive Volunteer Cyber Militia. 2011.

61. **Natalja Sleptšuk**. Investigation of the Intermediate Layer in the Metal-Silicon Carbide Contact Obtained by Diffusion Welding. 2011.

62. **Martin Jaanus**. The Interactive Learning Environment for Mobile Laboratories. 2011.

63. **Argo Kasemaa**. Analog Front End Components for Bio-Impedance Measurement: Current Source Design and Implementation. 2011.

64. **Kenneth Geers**. Strategic Cyber Security: Evaluating Nation-State Cyber Attack Mitigation Strategies. 2011.

65. **Riina Maigre**. Composition of Web Services on Large Service Models. 2011.

66. **Helena Kruus**. Optimization of Built-in Self-Test in Digital Systems. 2011.

67. **Gunnar Piho**. Archetypes Based Techniques for Development of Domains, Requirements and Sofware. 2011.

68. **Juri Gavšin**. Intrinsic Robot Safety Through Reversibility of Actions. 2011.

69. **Dmitri Mihhailov**. Hardware Implementation of Recursive Sorting Algorithms Using Tree-like Structures and HFSM Models. 2012.

70. **Anton Tšertov**. System Modeling for Processor-Centric Test Automation. 2012.

71. **Sergei Kostin**. Self-Diagnosis in Digital Systems. 2012.

72. **Mihkel Tagel**. System-Level Design of Timing-Sensitive Network-on-Chip Based Dependable Systems. 2012.

73. **Juri Belikov**. Polynomial Methods for Nonlinear Control Systems. 2012.

74. **Kristina Vassiljeva**. Restricted Connectivity Neural Networks based Identification for Control. 2012.

75. **Tarmo Robal**. Towards Adaptive Web – Analysing and Recommending Web Users` Behaviour. 2012.

76. **Anton Karputkin**. Formal Verification and Error Correction on High-Level Decision Diagrams. 2012.

77. **Vadim Kimlaychuk**. Simulations in Multi-Agent Communication System. 2012.

78. **Taavi Viilukas**. Constraints Solving Based Hierarchical Test Generation for Synchronous Sequential Circuits. 2012.

79. **Marko Kääramees**. A Symbolic Approach to Model-based Online Testing. 2012.

80. **Enar Reilent**. Whiteboard Architecture for the Multi-agent Sensor Systems. 2012.

81. **Jaan Ojarand**. Wideband Excitation Signals for Fast Impedance Spectroscopy of Biological Objects. 2012.

82. **Igor Aleksejev**. FPGA-based Embedded Virtual Instrumentation. 2013.

83. **Juri Mihhailov**. Accurate Flexible Current Measurement Method and its Realization in Power and Battery Management Integrated Circuits for Portable Applications. 2013.

84. **Tõnis Saar**. The Piezo-Electric Impedance Spectroscopy: Solutions and Applications. 2013.

85. **Ermo Täks**. An Automated Legal Content Capture and Visualisation Method. 2013.

86. **Uljana Reinsalu**. Fault Simulation and Code Coverage Analysis of RTL Designs Using High-Level Decision Diagrams. 2013.

87. **Anton Tšepurov**. Hardware Modeling for Design Verification and Debug. 2013.

88. **Ivo Müürsepp**. Robust Detectors for Cognitive Radio. 2013.

89. **Jaas Ježov**. Pressure sensitive lateral line for underwater robot. 2013.

90. **Vadim Kaparin**. Transformation of Nonlinear State Equations into Observer Form. 2013.

92. **Reeno Reeder**. Development and Optimisation of Modelling Methods and Algorithms for Terahertz Range Radiation Sources Based on Quantum Well Heterostructures. 2014.

93. **Ants Koel**. GaAs and SiC Semiconductor Materials Based Power Structures: Static and Dynamic Behavior Analysis. 2014.

94. **Jaan Übi**. Methods for Coopetition and Retention Analysis: An Application to University Management. 2014.

95. **Innokenti Sobolev**. Hyperspectral Data Processing and Interpretation in Remote Sensing Based on Laser-Induced Fluorescence Method. 2014.

96. **Jana Toompuu**. Investigation of the Specific Deep Levels in *p*-, *i*- and *n*-Regions of GaAs p^+-*pin*-n^+ Structures. 2014.

97. **Taavi Salumäe**. Flow-Sensitive Robotic Fish: From Concept to Experiments. 2015.

98. **Yar Muhammad**. A Parametric Framework for Modelling of Bioelectrical Signals. 2015.

99. **Ago Mõlder**. Image Processing Solutions for Precise Road Profile Measurement Systems. 2015.

100. **Kairit Sirts**. Non-Parametric Bayesian Models for Computational Morphology. 2015.

101. **Alina Gavrijaševa**. Coin Validation by Electromagnetic, Acoustic and Visual Features. 2015.

102. **Emiliano Pastorelli**. Analysis and 3D Visualisation of Microstructured Materials on Custom-Built Virtual Reality Environment. 2015.

103. **Asko Ristolainen**. Phantom Organs and their Applications in Robotic Surgery and Radiology Training. 2015.