A MODEL-FREE FRACTIONAL-ORDER PID TUNING METHOD BASED ON AN IMPROVEMENT OF PARTICLE SWARM OPTIMIZATION

by

Deniz Seviş

B.S., in Electrical and Electronics Engineering, Pamukkale University, 2007

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Systems and Control Engineering

Boğaziçi University

2010

ACKNOWLEDGEMENTS

I would like to express my gratitude to all people who have helped and inspired me during my study.

I would like to give my special thanks to my supervisor Yağmur Denizhan. The preparation of this thesis would not have been possible without her guidance, encouragement and endless efforts. Her perpetual energy and enthusiasm in research had motivated me. In addition, she was always accessible and willing to help with my thesis. Therefore, research life became smooth and rewarding for me.

I am thankful to my undergraduate supervisor Serdar İplikçi for being such an encouraging advisor, advices for my career, and providing the chance of meeting my colleagues and my supervisor Yağmur Denizhan.

I want to thank Kamil Şenel and Gökhan Habiboğlu for their support with their programming skills, friendship, and intangible support. All were precious for me.

My deepest gratitude goes to my family for their unflagging love and support throughout my life. I am indebted to my family their care and love. I have no suitable word that can fully describe my gratitude to them.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS/ABBREVIATIONS

a_i	Coefficients of fractional-order transfer function's denominator
c_1	Cognitive learning rate
c_2	Social learning rate
D^α	Fractional-order derivative operator
$e(t)$	Tracking error
$E_\alpha(z)$	One-parameter Mittag-Leffler Function
$E_{\alpha,\beta}(z)$	Two-parameter Mittag-Leffler Function
f	Fitness function
$G_c(s)$	Controller transfer function
$G_p(s)$	Plant transfer function
$g_t(\tau)$	The relationship between real time and cosmic time
h	Time-step for numerical solution of fractional-order differential equations
J^α	Fractional-order integral operator
K_{cr}	Critical gain that oscillation is observed
K_P	Proportional gain of PID controller
K_I	Integral gain of PID controller
K_D	Proportional gain of PID controller
L	Memory length
$L\{.\}$	Laplace transform operator
M_p	Maximum overshoot
P_i	Personal best of i^{th} particle
P_g	Global best
P_{reflex}	Probability of switching to Reflex-B mode from socio-cognitive mode
$R(s)$	Laplace transform of reference input signal
$S_o(t)$	Real distance passed by an object
T	Cosmic time

T_{cr}	Critical period of oscillation
t_r	Rise-time
t_s	Settling-time
u_m	Control signal
V_i	Current velocity of i^{th} particle
$v(t)$	Individual velocity of an object
$v_o(t)$	Velocity of an object from the view point of an independent observer
w_k	Inertia factor at k^{th} iteration
X_i	i^{th} particle
y_m	Closed-loop system output
α	Order of fractional-order derivative
ε_{M_p}	Percentage error of overshoot based on its reference value
ε_{ss}	Percentage error of rise-time based on its reference value
ε_{ss}	Percentage error of settling-time based on its reference value
ε_{ss}	Percentage steady-state error
$\Gamma(.)$	Gamma function
ϕ	Stability boundary
φ_1	Random function
φ_2	Random function
λ	Order of the integral action of fractional-order PID
μ	Order of the derivative action of fractional-order PID
τ	Individual time of an object

BC	Brunch cuts
GA	Genetic Algorithm
GL	Grünwald-Letnikov
LTI	Linear time-invariant
KS-PSO	Knowledge-Supported Particle Swarm Optimisation
PSO	Particle Swarm Optimisation
RHP	Right half plane
RL	Riemann-Liouville

ML	Mittag-Leffler
$PI^{\lambda}D^{\mu}$	Fractional-order PID controller

1. INTRODUCTION

The theory of fractional derivative has been carried on for three centuries. However, in recent times it is rather the applications of Fractional Calculus that have gained importance in various areas of science and engineering. Fractional derivatives seem to be adequate for describing memory and hereditary properties for a large field of materials and processes. There are various application areas of Fractional Calculus which lead themselves better to a representation in terms of fractional-order equations. Fractional *Diffeointegral* equations and fractional calculus applications include Fluid Flow, Rheology, Dynamical Processes in Self-Similar and Porous Structures, Diffusive Transport Akin to Diffusion, Electrical Networks, Probability and Statistics, Control Theory of Dynamical Systems, Viscoelasticity, Electrochemistry of Corrosion, Chemical Physics, Optics and Signal Processing etc. [1].

In the realm of control engineering Fractional Calculus started to draw the attention of researchers a decade ago with its most common application, namely Fractional-Order PID control, introduced first by [2]. In [3] a modified root locus method is presented for fractional-order systems and fractional-order controllers.

The most basic form of Fractional Order PID controller, referred to as $PI^{\lambda}D^{\mu}$ controller differs from its integer order counterpart by the fact that the orders of integral and derivative actions are not one but parameters that can take on real values. This increase in the number of parameters -while providing extra degrees of freedom for achieving the design specifications- renders most of the traditional PID tuning methods inapplicable. The fundamental difficulty in tuning λ and μ is due to the fact that there exists no explicit monotonous relation between these parameters and performance criteria like overshoot (M_p), rise-time (t_r), and settling-time (t_s). Furthermore, as shown in [4] by qualitative analysis and simulations, the closed-loop behavior of the system exhibits very sensitive dependence on μ in particular. These facts suggest that the tuning problem of λ and μ would lend itself better to derivative-free stochastic optimization techniques rather than deterministic search methods.

Optimization is the most common problem not only of all engineers and scientists but of all living beings. Irrespective of the area, any optimization method can be considered as a weighted combination of two basic search approaches: Exploitation of systematic knowledge about the system at hand and blind search. The former approach has a context and problem-specific character because any systematic knowledge of reasonable extent is typically valid within a rather limited context under very specific conditions. Provided that the system at hand satisfies such conditions, and that systematic knowledge about it is available, optimization methods dominated by the former approach yield fast and reliable solutions. Analytical model-based optimization methods typically fall into this category. But also model-free methods that assume some specific properties (such as gradient-based optimization methods that assume a certain smoothness of the objective function) can be considered within this category. Blind search approach, on the other hand, relies on stochastic methods and can be performed without much knowledge about the specific area of application. This advantage is, however, usually accompanied by some kind of volatility in terms of search performance.

Starting from the last decades of the 20^{th} century the trend of biologically-inspired computational approaches brought forth many heuristic optimization techniques. Among the most popular ones one can count evolutionary algorithms and swarm algorithms.

This study proposes a model-free fractional-order tuning method which is Knowledge-Supported Particle Swarm Optimization technique, suggesting a different combination of classical PID tuning approaches and stochastic optimization approaches. The stochastic method will be based on the so-called Particle Swarm Optimization (PSO) Technique, which is becoming a popular design method ever since it was introduced 1995 in [5]. PSO has already been used for approximation of fractional-order systems [6] as well as for $PI^{\lambda}D^{\mu}$ controller design [7]. The innovation in this study is an improvement in this technique that combines the "blind search" inherent to PSO with "knowledge-guided search" in cases where some systematic knowledge about the optimization problem is available.

1.1. Definition of the Problem

The most common application of Fractional Calculus in control theory is the fractional-order variant of the classical PID controller. This generalization of PID controller to fractional-order derivatives and integrals was first introduced by [2]. The basic problem in both integer and fractional-order PID controllers is the proper choice of its parameters for a given system. For integer-order PID controller various methods exist in the literature for a long time.

The integer-order PID tuning methods can in general be categorized into one of the two main groups: (i) Model-based, (ii) Model-free.

(i) Model-based: Model-based design method requires modelling the system, analysis and synthesis of the controller for the plant. Model-based design provides an efficient approach for establishing a controller for a system, but modeling and designing a controller based on system's mathematical model is difficult and time consuming.

(ii) Model-free: Model-free control is based on the knowledge of the input-output behavior of a system, i.e., there is no need of mathematical model of the system. Any identification procedure is not required to design a controller that stabilizes the system.

Some parameter tuning methods have been proposed also for fractional-order PID (i.e. $PI^{\lambda}D^{\mu}$) controllers since the concept had been introduced into the literature. They can also be categorized in a similar manner.

The purpose of this thesis is to develop a model-free $PI^{\lambda}D^{\mu}$ tuning algorithm and that seeks for a set of parameters $(K_p, K_i, K_d, \lambda, \mu)$ which guarantees achievement of a given set of time-domain design specifications in terms of overshoot (M_p), rise-time (t_r), and settling-time (t_s). The innovation in this study is an improvement in PSO technique that combines the "blind search" inherent to PSO with "knowledge-supported search" in cases where some systematic knowledge about the optimization

problem is available. With the help of this systematic knowledge and initialization of PID controller's integral and derivative actions' order with integer numbers, the algorithm can give priority to integer order PID controller. Unless integer-order PID controller can be found between the allowable limits, the algorithm seeks for fractional-order PID controller parameters that meet the design specifications.

1.2. Thesis Outline

The first chapter gives brief introductory notes on fractional calculus and fractional-order PID controllers, and defines the problem dealt with in this study.

Second chapter gives information about Fractional Calculus.

In the third chapter, stability analysis of fractional-order systems is described based on the comparison with integer-order system stability analysis.

In the fourth chapter, information about fractional-order PID controller is given, then parameter tuning methods in the literature for both integer and fractional-order PID controllers follows in chapter 5.

Chapter 6 describes the proposed model-free $PI^{\lambda}D^{\mu}$ tuning method in this study.

Performance of the proposed method are illustrated via examples in chapter 7.

Finally, in chapter 8, results are discussed and the thesis is concluded.

2. FRACTIONAL CALCULUS

The traditional integral and derivative have significant importance for the technology as the way of understanding and working with natural and artificial systems. Fractional Calculus is a field of mathematics that grew out of the traditional definitions of the classical integral and derivative operators. It is the branch of calculus that generalizes the derivative and integral to non-integer orders. Therefore, it is called *generalized diffeontegrals.*

Fractional calculus is three centuries old as the conventional calculus, but not very popular among science and/or engineering community. The advantage of this subject is that fractional derivatives and integrals are not local (or point) operators. Fractional derivatives and integrals provide an excellent tool for the description of memory properties. Thereby, this subject translates the reality of nature better! In order to make this subject popular in science and engineering community, a new definition called *diffeointegration* was developed. Diffeointegral is a new dimension to understand or describe basic nature in a better way which is an operator doing both differentiation and integrations, in a general sense [8].

In this study, fractional order derivatives and integrals are limited to real numbers; the complex order diffeointegrals are not covered.

2.1. Historical Background

Fractional Calculus, born out of L'Hôpital's question to Leibnitz about the possibility of $n = \frac{1}{2}$ in the n^{th} order derivative notation $\frac{d^n y}{dx^n}$, is almost as old as classical (integer order) calculus itself.

Fractional Calculus took attention of pioneering mathematicians of its era - Leibniz, Bernoulli, and Euler- in time. Leibnitz mentions derivatives of general order in his correspondence with Johann Bernoulli. Leibnitz used the notation of "$d^{\frac{1}{2}}y$" to

denote the derivative of order $\frac{1}{2}$.

In 1812 Laplace defined a fractional derivative by means of an integral and in 1819,starting with $y = x^m$, m is a positive integer, Lacroix developed the n^{th} order derivative as follows:

$$\frac{d^n y}{dx^n} = \frac{m!}{(m-n)!} x^{m-n} \tag{2.1}$$

By using Legendre's symbol for generalized factorial which is the Gamma function (see: Equation (2.12)), he obtained the following:

$$\frac{d^n y}{dx^n} = \frac{d^{1/2}(x^m)}{dx^{1/2}} = \frac{\Gamma(m+1)}{\Gamma(m-n+1)} x^{m-n} \tag{2.2}$$

Followed from the Equations (2.1) and (2.2), the derivative of order 1/2 was obtained for $y = x^m$ when m is one is as follows [9]:

$$\frac{d^{1/2} y}{dx^{1/2}} = \frac{\Gamma(2)}{\Gamma(3/2)} x^{1/2} = \frac{2}{\sqrt{\Pi}} \sqrt{x} \tag{2.3}$$

This result obtained by Lacroix is the same as the present-day's result obtained by using the Riemann-Liouville definition of a fractional derivative.

In 1822, Joseph Fourier obtained fractional order derivative of arbitrary order from his integral representation of $f(x)$

$$f(x) = \frac{1}{2\Pi} \int_{-\infty}^{\infty} f(u) du \int_{-\infty}^{\infty} \cos p(x-u) dp \tag{2.4}$$

$$\frac{d^n}{dx^n} \cos p(x-u) = p^n \cos[p(x-u) + \frac{1}{2} n\Pi] \tag{2.5}$$

when n is an integer. If we replace n with an arbitrary number α, then the generalized

formulation is as follows:

$$\frac{d^\alpha}{dx^\alpha} f(x) = \frac{1}{2\Pi} \int_{-\infty}^{\infty} f(u)du \int_{-\infty}^{\infty} p^\alpha \cos p(x-u)dp \tag{2.6}$$

Fractional calculus had not been studied for almost a decade until Joseph Liouville published his works. He published three recollections and several publications in 1832. He started his developments for fractional calculus with integer order derivatives:

$$D^n e^{ax} = a^n e^{ax}, \tag{2.7}$$

which was extended naturally to arbitrary order derivative:

$$D^\alpha e^{ax} = a^\alpha e^{ax}, \tag{2.8}$$

He assumed that the derivative of a function of arbitrary order may be expanded in a series of the form:

$$f(x) = \sum_{m=0}^{\infty} c_m e^{a_m x} \tag{2.9}$$

is

$$D^\alpha f(x) = \sum_{m=0}^{\infty} c_m a_m^\alpha e^{a_m x} \tag{2.10}$$

This formula is known as Liouville's first formula for fractional order derivative. It allowed a generalization of α, the order of the derivative, from a mere integer to any number: rational, irrational, or complex. G. F. Bernhard Riemann developed fractional integration theory and it was published in 1892. He derived the generalization of Taylor series as follows:

$$D^\alpha f(x) = \frac{1}{\Gamma(\alpha)} \int_c^x (x-t)^{\alpha-1} f(t)dt + \Psi(x) \tag{2.11}$$

$\Psi(x)$ is a complementary function to provide a measure of the deviation from the law of exponents. In 1892 Oliver Heaviside published papers where he mentioned that linear differential equations may be solved by using generalized operators. Limited number of work was published about fractional calculus during the period 1900-1970. The first international conference was held at the University of New Heaven in 1974.

Significant number of mathematical activities on the subject of fractional calculus were developed in Japan in the 1980's.

Most of the mathematical theory applicable to fractional calculus was developed prior to the turn of the 20th century. However, the most interesting studies on engineering and scientific applications have been found in the past 100 years. The Mathematics had to change to meet some requirements of physical realities in some cases. Caputo reformulated the classic Riemann-Liouville fractional derivative definition in order to use integer order initial conditions to solve fractional order differential equations [10].

2.2. Special Functions of Fractional Calculus

In order to understand the idea and use of Fractional Calculus, some mathematical definitions that arise in this concept need to be discussed. The Gamma function and the Mittag-Leffler function play the most important role in fractional calculus and fractional order differential equation theory. There are various functions employed in Fractional Calculus instead of Gamma Function and Mittag-Leffler Function which are given in the appendix.

2.2.1. Gamma Function

One of the basic functions of fractional calculus is Euler's gamma function $\Gamma(z)$ which is the generalization of integer order factorial n!, and it allows to choose n both non-integer and complex values [10].

The Gamma function is defined by the integral

$$\Gamma(z) = \int_0^\infty e^{-u} u^{z-1} du \tag{2.12}$$

which converges to the right half of the complex plane $Re(z) > 0$.

2.2.1.1. Some Properties of The Gamma Function. One of the basic properties of the Gamma function is as follows:

$$\Gamma(z+1) = z\Gamma(z) \tag{2.13}$$

It can be easily obtained that $\Gamma(1) = 1$, and by using the property defined by Equation (2.13) the Gamma function values for z=2,3,... can be obtained as follows:

$$\begin{aligned}
\Gamma(2) &= 1\Gamma(1) = 1 = 1!, \\
\Gamma(3) &= 2\Gamma(2) = 2 \times 1! = 2!, \\
\Gamma(4) &= 3\Gamma(3) = 3 \times 2! = 3!, \\
&\vdots \\
\Gamma(n+1) &= n\Gamma(n) = n(n-1)! = n!
\end{aligned}$$

A consequence of this property is that the Gamma function gives the same values as the factorial operation when n is an integer number.

Another important property of the Gamma function is that it has simple poles at negative integer numbers ($z = -n, n = 1, 2, ...$). Figure 2.1 demonstrates the Gamma function values at and around zero. It must be noted that the Gamma function goes to infinity at negative integer values, but it is still defined at non-integer values.

Generalization of the factorial operation with the Gamma function allows to

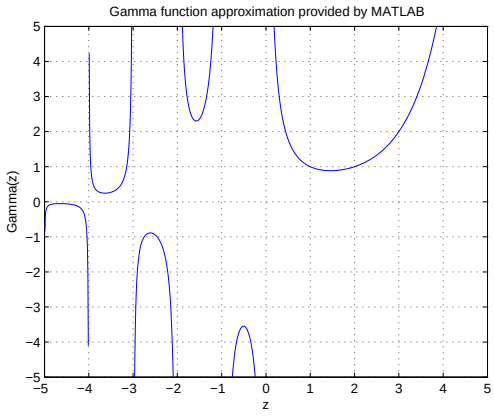

Figure 2.1. MATLAB approximation of Gamma function

compute the following binomial operation:

$$\binom{n}{m} = \frac{n!}{m!(n-m)!} = \frac{\Gamma(n+1)}{\Gamma(m+1)\Gamma(n-m+1)} \tag{2.14}$$

which is also valid for positive integer values of n and m, as well as real values.

In some cases using beta function instead of certain combinations of gamma function is more convenient. Definition of beta function is given in the appendix.

2.2.2. Mittag-Leffler Function

The exponential function has a significant importance in integer order differential equations. Since the exponential function is a special form of the Mittag-Leffler function, the Mittag-Leffler function plays an important role in the solution of non-integer order differential equations.

One-parameter generalization of this function is denoted by:

$$E_\alpha(z) = \sum_{k=0}^{\infty} \frac{z^k}{\Gamma(\alpha k + 1)}, \tag{2.15}$$

was introduced by G. M. Mittag-Leffler in 1903 [10].

The two-parameter function of Mittag-Leffler type is defined as follows:

$$E_{\alpha,\beta}(z) = \sum_{k=0}^{\infty} \frac{z^k}{\Gamma(\alpha k + \beta)} \tag{2.16}$$

where $\alpha > 0, \beta > 0$.

The identities and relations to other functions of Mittag-Leffler Function is given in the appendix.

2.3. Fractional Order Operators

In this section several approaches to generalization of fractional order differentiation and integration are considered. Although *fractional order* is actually a misnomer, we will still adhere to this commonly used terminology to designate derivatives or integrals of arbitrary order. Fractional-Order Derivative (Integral) is the commonly-used name for the derivative (integral) of arbitrary order.

2.3.1. Fractional-Order Derivative and Integral Definitions

The classical differentiation operator is $D = d/dx$, and if n is a positive integer, $D^n f(x)$ means the n^{th} derivative of $f(x)$, but in the theory of Fractional calculus, n can be any arbitrary number.

If the infinite sequence of n-fold integrals and n-fold derivatives are considered,

$$\cdots, \int_a^t d\tau_2 \int_a^{\tau_2} f(\tau_1) d\tau_1, \quad \int_a^t f(\tau_1) d\tau_1, \quad f(t), \quad \frac{df(t)}{dt}, \quad \frac{d^2 f(t)}{dt^2}, \cdots \tag{2.17}$$

the derivative of arbitrary real order α can be considered as an interpolation of this sequence of operators [10]. Suppose that $Re\{\alpha\} > 0$ and let n be the smallest integer

greater than $Re\{\alpha\} > 0$, and $\nu = n - \alpha$. Then,

$$0 < Re\{\nu\} \leq 1 \tag{2.18}$$

The fractional derivative of $f(x)$ of order α is defined as follows:

$${}_cD_x^{\alpha} f(x) = {}_cD_x^{n}[{}_cD_x^{\nu} f(x)] \tag{2.19}$$

for $x > 0$.

The following notation will be used for fractional-order derivative which was suggested and used by Davis [11]:

$${}_aD_t^{\alpha} f(t) \tag{2.20}$$

The subscripts a and t denote the lower and upper limits of the operation of fractional differentiation. Ross called those limits the *terminals* of fractional differentiation in [9].

There exists various different definitions for fractional-order derivative and fractional-order integral in the literature. Riemann-Liouville, Caputo, and Grünwald-Letnikov definitions are the most common ones among them.

2.3.1.1. Riemann-Liouville Definition. Riemann-Liouville (RL) definition is the most used generalization of the derivative and is based on Cauchy's integral (appendix) formula. RL derivative with the lower limit of integration is defined as follows:

$${}_aD_t^{\alpha} f(t) = \frac{1}{\Gamma(n-\alpha)} \left(\frac{d}{dt}\right)^n \int_a^t \frac{f(\tau)}{(t-\tau)^{\alpha-n+1}} d\tau, \tag{2.21}$$

$$n - 1 \leq \alpha < n$$

where n is integer and α is a real number. It is enough to require integrability of the function $f(t)$ for proper use of the RL definition of fractional derivative.

If $n-1 \leq \alpha < n$, then the left-sided RL derivative is defined by:

$$ {}_aD_t^{\alpha} f(t) = \frac{1}{\Gamma(n-\alpha)} \left(\frac{d}{dt}\right)^n \int_a^t (t-\tau)^{n-\alpha-1} f(\tau) d\tau \tag{2.22} $$

The corresponding right-sided RL derivative is as follows:

$$ {}_tD_b^{\alpha} f(t) = \frac{1}{\Gamma(k-\alpha)} \left(-\frac{d}{dt}\right)^k \int_t^b (\tau-t)^{k-\alpha-1} f(\tau) d\tau \tag{2.23} $$

RL definition of fractional integral can be obtained by the generalization of the n-fold integration by replacing the factorial operation for integer numbers with the Gamma function for any real number α as given in the Equation (2.24):

$$ \begin{aligned} D^{-n} f(t) = J^n f(t) &= \frac{1}{(n-1)!} \int_0^t (t-\tau)^{n-1} f(\tau) d\tau \\ D^{-\alpha} f(t) = J^{\alpha} f(t) &= \frac{1}{\Gamma(\alpha)} \int_0^t (t-\tau)^{\alpha-1} f(\tau) d\tau \end{aligned} \tag{2.24} $$

2.3.1.2. Caputo Definition. The Caputo definition of the fractional derivative is as follows:

$$ {}_a^C D_t^{\alpha} f(t) = \frac{1}{\Gamma(\alpha-n)} \int_a^t \frac{f^{(n)}\tau}{(t-\tau)^{\alpha+1-n}} d\tau, \qquad n-1 \leq \alpha < n \tag{2.25} $$

The main advantage of the Caputo's approach is that the initial conditions for the fractional differential equations take the same form as for integer-order differential

equations.

One of the differences between the RL definition and Caputo definition of fractional derivative is that Caputo derivative of a constant is zero whereas the RL derivative of a constant is not.

2.3.2. Fractional-Order Diffeointegral

Differintegration operator does both integration and differentiation. Differentiation can easily be turned into integration by only changing the sign of the derivative order positive to negative.

2.3.2.1. Grünwald-Letnikov Definiton. If we consider the definition of the n^{th}-order derivative of a function $f(t)$ as:

$$f^{(n)}(t) = \frac{d^n f}{dt^n} = \lim_{h \to 0} h^{-n} \sum_{m=0}^{n} (-1)^m \binom{n}{m} f(t - mh) \tag{2.26}$$

Then, the generalized form of 2.26 is defined as:

$$D_h^{(\alpha)} f(t) = \lim_{h \to 0} h^{-\alpha} \sum_{m=0}^{\frac{t-a}{h}} (-1)^m \binom{\alpha}{m} f(t - mh) \tag{2.27}$$

or equivalently

$$D_h^{(\alpha)} f(t) = \lim_{n \to \infty} (\frac{n}{t - \alpha}) \sum_{m=0}^{n} (-1)^m \binom{\alpha}{m} f(t - m(\frac{t - \alpha}{n})) \tag{2.28}$$

where

$$\binom{\alpha}{m} = \frac{\Gamma(\alpha + 1)}{m! \Gamma(\alpha - m + 1)} \tag{2.29}$$

and n is the memory length. For the negative values of α the process becomes integration:

$$D_h^{(-\alpha)} f(t) = \lim_{h \to 0} h^{\alpha} \sum_{m=0}^{\frac{t-a}{h}} \frac{\Gamma(\alpha + m)}{m! \Gamma(\alpha)} f(t - mh) \tag{2.30}$$

2.3.3. Some Properties of Fractional-Order Operators

2.3.3.1. Linearity. Similar to the integer-order differentiation, fractional-order differentiation is a linear operation:

$$D_{\alpha}(\lambda f(t) + \mu g(t)) = \lambda D_{\alpha} f(t) + \mu D_{\alpha} g(t) \tag{2.31}$$

where D_{α} denotes any fractional-order derivative.

2.3.3.2. The Leibniz Rule. The Leibniz rule for fractional-order derivative is as follows: If $f(\tau)$ is continuous in $[a, t]$ and $g(\tau)$ has $n + 1$ continuous derivatives in $[a, t]$, then the fractional derivative of the production of the two functions $g(\tau) f(\tau)$ is defined as:

$${}_a D_t^{\alpha}(g(\tau) f(\tau)) = \sum_{k=0}^{\infty} \binom{\alpha}{k} g^{(k)}(t) {}_a D_t^{\alpha - k} f(t) \tag{2.32}$$

The Leibniz formula for fractional integral of the production of two functions of order α is defined as:

$$D^{-\alpha}(f(t) g(t)) = \sum_{k=0}^{+\infty} \binom{-\alpha}{k} (D^k g(t))(D^{-\alpha - k} f(t)) \tag{2.33}$$

2.4. Fractional-Order Systems

Contrary to the traditional approach, fractional-order systems have transfer functions of arbitrary real-order. According to the definition in [2] *Fractional-Order System* means "systems which are better described by fractional-order mathematical models".

In order to work with fractional-order systems and fractional-order transfer functions, Generalized Laplace Transform should be considered.

2.4.1. Generalized Laplace Transform

Fractional-order differential equations appear more frequently in various research and engineering applications. Laplace transform technique is an effective method for finding solutions to both integer-order and fractional-order differential equations.

The function $F(s)$ of the complex variable s is defined as

$$F(s) = L\{f(t)\} = \int_0^\infty e^{-st} f(t) dt \tag{2.34}$$

is called the Laplace transform of function $f(t)$.

The original function $f(t)$ can be obtained by the inverse Laplace transform

$$f(t) = L^{-1}\{F(s)\} = \int_{c-i\infty}^{c+i\infty} e^{st} F(s) ds, \quad c = Re(s) > c_0, \tag{2.35}$$

where c_0 lies in the right half plane of the absolute convergence of the Laplace integral.

The Laplace transform of the convolution

$$f(t) * g(t) = \int_0^t f(t-\tau)g(\tau) = \int_0^t f(\tau)g(t-\tau)d\tau \tag{2.36}$$

is equal to the product of the Laplace transform of those functions:

$$L\{f(t) * g(t)\} = F(s)G(s) \tag{2.37}$$

Another useful property for the generalization of the Laplace transform is the Laplace transform of the derivative of a function $f(t)$ of order n is:

$$L\{f^n(t)\} = s^n F(s) - \sum_{k=0}^{n-1} s^{n-k-1} f^{(k)}(0) = s^{(n)} F(s) - \sum_{k=0}^{n-1} s^k f^{(n-k-1)}(0) \tag{2.38}$$

2.4.1.1. Laplace Transform of The Riemann-Liouville Fractional Derivative. The Laplace transform of the Riemann-Liouville derivative of order α is as follows:

$$L\{{}_0D_t^\alpha f(t)\} = s^\alpha F(s) - \sum_{k=0}^{n-1} s^k [{}_0D_t^{\alpha-k-1} f(t)]_{t=0} \tag{2.39}$$
$$n-1 \leq \alpha < n$$

2.4.1.2. Laplace Transform of The Caputo Fractional Derivative. The Laplace transform of the Caputo fractional derivative of order α is as follows:

$$L\{{}_0D_t^\alpha f(t)\} = s^\alpha F(s) - \sum_{k=0}^{n-1} s^{\alpha-k-1} f^k(0) \tag{2.40}$$
$$n-1 \leq \alpha < n$$

2.4.1.3. Laplace Transform of The Grünwald-Letnikov Fractional Differintegral. Assuming that $n < \alpha < n+1$ the Laplace transform of the Grunwald-Letnikov fractional differintegral of order α is defined as:

$$\begin{aligned} L\{{}_0D_t^{\alpha} f(t)\} &= \sum_{k=0}^{n} f^{(k)}(0) L\{\frac{t^{-\alpha+k}}{\Gamma(-\alpha+n+1)}; s\} \\ &+ L\{\frac{t^{n-\alpha}}{\Gamma(-\alpha+n+1)} * f^{(n+1)}(t); s\} \\ &= \sum_{k=0}^{n} f^{(k)}(0) s^{\alpha} - k - 1 \\ &+ s^{\alpha} - n - 1(s^{n+1}F(s) - \sum_{k=0}^{n} f^{(k)}(0) s^{n-k}) \\ &= s^{\alpha}F(s) \end{aligned} \tag{2.41}$$

This definition is corresponds to Grünwald-Letnikov fractional derivative since the order α is positive. If α is considered as a negative real number then the Laplace transform definition of Grünwald-Letnikov fractional derivative turns into the Laplace transform of Grünwald-Letnikov fractional integral.

2.4.1.4. Laplace Transform of Fractional-Order Integral. Laplace transform of fractional-order integral which can be easily derived by using the Laplace transform of the convolution is as follows:

$$L\{I^{\alpha} f(t)\} = s^{-\alpha}F(s) \tag{2.42}$$

This definition is valid both for Riemann-Liovuville and Grünwald-Letnikov fractional integral.

2.5. Fractional-Order Differential Equations

Even solving a simple linear differential equation requires substantial effort. Laplace transform method is a popular technique for solving fractional-order differential equa-

tions. Numerical evaluation with short-memory principle, Mellin transform, power series expansion method using fractional Green's function, Babenko's symbolic method, orthogonal polynomial method, Reisz fractional potential method, method with Wright's function, and finite-part integral method are some of the mathematical tools for obtaining the solutions of fractional-order differential equations [10].

Fractional-order dynamical systems are described with fractional-order differential equations. Both linear and nonlinear fractional-order differential equations appear in applications in many fields of science. Known methods for finding solutions of integer-order differential equations do not work in the case of fractional-order differential equations. There exist several methods for finding analytical solution of fractional-order differential equations in the literature. Podlubny has solved fractional-order differential equations consisted of derivatives in Riemann-Liouville definition by using Laplace transform method in [12], but this solution method is only valid for fractional-order differential equations with constant coefficients. Hashim and Abdulaziz obtained exact and approximate solutions for both linear and nonlinear fractional-order differential equations consisted of derivatives in Caputo sense by Homotopy analysis [13]. There are more methods as in [14], [15], and [16].

In this study, the solution of linear fractional differential equations with constant coefficients is considered. Because of its simplicity and effectiveness, Laplace transform method is explained for the solution method for fractional-order differential equations.

2.5.1. Analytical Solution of Fractional-Order Differential Equations

There are various methods for finding analytical solutions of fractional-order differential equations and most of them are in the form of infinite series.

As mentioned before, Mittag-Leffler function is the generalization of the exponential function and it plays an important role in the solution of fractional-differential equations. Since the solutions found by the Laplace transform method are described in terms of the Mittag-Leffler function, the Laplace transform of the Mittag-Leffler

function is needed. First, it should be proved that,

$$\int_0^\infty e^{-t}e^{\pm zt}dt = \frac{1}{1\mp z}, \quad |z|<1. \tag{2.43}$$

By using the series expansion for e^z, it is obtained that,

$$\int_0^\infty e^{-t}e^{zt}dt = \frac{1}{1-z} = \sum_{k=0}^{\infty}\frac{(\pm z)^k}{k!}\int_0^\infty e^{-t}t^k dt = \sum_{k=0}^{\infty}(\pm z)^k = \frac{1}{1\mp z} \tag{2.44}$$

Second, both sides of Equation (2.43) is differentiated with respect to z:

$$\int_0^\infty e^{-t}t^k e^{\pm zt}dt = \frac{k!}{(1-z)^{k+1}}, \quad (|z|<1) \tag{2.45}$$

After making the necessary substitutions, the Laplace transform of the function $t^k e^{\pm at}$ is obtained:

$$\int_0^\infty e^{-pt}t^k e^{\pm at}dt = \frac{k!}{(p\mp a)^{k+1}}, \quad (Re(p)>|a|) \tag{2.46}$$

Substitution of the two-parameter Mittag-Leffler function in the integral leads to:

$$\int_0^\infty e^{-t}t^{\beta-1}E_{\alpha,\beta}(zt^\alpha)dt = \frac{1}{1-z}, \quad (|z|<1). \tag{2.47}$$

From Equation (2.47) a pair of Laplace transforms of the function $t^{\alpha k+\beta-1}E^{(k)}_{\alpha,\beta}(\pm zt^\alpha)$, $(E^{(k)}_{\alpha,\beta}(y) \equiv \frac{d^k}{dy^k}E_{\alpha,\beta}(y))$: Substitution of the two-parameter Mittag-Leffler function in

the integral leads to:

$$\int_0^\infty e^{-pt} t^{\alpha k+\beta-1} E^{(k)}_{\alpha,\beta}(\pm a t^\alpha) dt = \frac{k! p^{\alpha-\beta}}{(p^\alpha \mp a)^{k+1}}, \quad (Re(p) > |a|^{\frac{1}{\alpha}}) \tag{2.48}$$

The particular case of Equation (2.48) for $\alpha = \beta = \frac{1}{2}$ is useful for solving semidifferential equations:

$$\int_0^\infty e^{-pt} t^{\frac{k-1}{2}} E^{(k)}_{\frac{1}{2},\frac{1}{2}}(\pm a\sqrt{t}) dt = \frac{k!}{(\sqrt{p} \mp a)^{k+1}}, \quad (Re(p) > |a|^2) \tag{2.49}$$

The following examples from [10] illustrate the use of Equation (2.47) for solving fractional differential equations with constant coefficients.

Example 1: Assume that a fractional order differential equation is given as follows:

$${}_0D_t^{1/2} f(t) + a f(t) = 0, \quad (t > 0); \qquad [{}_0D_t^{-1/2} f(t)]_{t=0} = C \tag{2.50}$$

By applying the Laplace transform, we get:

$$F(s) = \frac{C}{s^{1/2} + a} \tag{2.51}$$

The solution of the fractional-order differential equation which is the inverse Laplace transform of $F(s)$ can easily be obtained with the help of Equation (2.49):

$$f(t) = C t^{-1/2} E_{\frac{1}{2},\frac{1}{2}}(-a\sqrt{t}) \tag{2.52}$$

Example 2: In this example the fractional differential equation includes more than one fractional-order derivative. The fractional-order differential equation is given

as:

$$ {}_0D_t^Q f(t) +_0 D_t^q f(t) = h(t) \tag{2.53} $$

Suppose that $0 < q < Q < 1$. The Laplace transform of the differential equation leads to

$$ \begin{aligned} (s^Q + s^q)F(s) &= C + H(s), \\ C &= [{}_0D_t^{q-1} f(t) +_0 D_t^{Q-1} f(t)]_{t=0}, \end{aligned} \tag{2.54} $$

and then

$$ F(s) = \frac{C + H(s)}{s^Q + s^q} = \frac{C + H(s)}{s^q(s^{Q-q} + 1)} = (C + H(s))\frac{s^{-q}}{s^{Q-q} + 1} \tag{2.55} $$

After inversion with the help of Equation (2.48) for $\alpha = Q - q$ and $\beta = Q$, the solution is obtained as:

$$ f(t) = CG(t) + \int_0^t G(t - \tau)h(\tau)d\tau, \tag{2.56} $$

$$ C = [{}_0D_t^{q-1} f(t) +_0 D_t^{Q-1} f(t)]_{t=0}, \quad G(t) = t^{Q-1} E_{Q-q,Q}(-t^{Q-q}) \tag{2.57} $$

Let's consider another example which includes sequential fractional-order derivatives:

Example 3:

$$ {}_0D_t^{\alpha}({}_0D_t^{\beta} y(t)) +_0 D_t^q y(t) = h(t), \tag{2.58} $$

where $0 < \alpha < 1,\ 0 < \beta < 1,\ 0 < q < 1,\ \alpha + \beta = Q > q$

The Laplace transform of the differential equation is:

$$(s^{\alpha+\beta}+s^{q})Y(s)=H(s)+s^{\alpha}b_2+b_1, \tag{2.59}$$

$$b_1=[{}_0D_t^{\alpha-1}({}_0D_t^{\beta}y(t))]_{t=0}+[{}_0D_t^{q-1}y(t)]_{t=0},$$

$$b_2=[{}_0D_t^{\beta-1}y(t)]_{t=0} \tag{2.60}$$

By writing $Y(s)$ in the form of

$$Y(s)=\frac{s^{-q}H(s)}{s^{\alpha+\beta-q}+1}+b_2\frac{s^{\alpha-q}}{s^{\alpha+\beta-q}+1}+b_1\frac{s^{-q}}{s^{\alpha+\beta-q}+1} \tag{2.61}$$

and finding the inverse Laplace transform with the help of Equation (2.48), the solution can be obtained as:

$$y(t)=b_2t^{\beta-1}E_{\alpha+\beta-q,\beta}(-t^{\alpha+\beta-q})+b_1t^{\alpha+\beta-q}E_{\alpha+\beta-q,\alpha+\beta}(-t^{\alpha+\beta-q})$$
$$+\int_0^t(t-\tau)^{\alpha+\beta-1}E_{\alpha+\beta-q,\alpha+\beta}(-(t-\tau)^{\alpha+\beta-q})h(\tau)d\tau. \tag{2.62}$$

2.5.2. Numerical Solution of Fractional-Order Differential Equations

Generally, it is not guaranteed that analytical solutions of fractional-order differential equations can be found. Numerical solutions of fractional-order differential equations can be found by means of existing methods in the literature. Finding numerical solutions can be considered as simulating the dynamical system that corresponds to that fractional-order differential equation.

Simulating fractional-order continuous-time systems is much more difficult than simulating integer-order systems. The simulation of fractional-order derivatives and integrals involves various computational problems such as extensive memory occupation, long computation time, etc. Development of numerical methods for fractional-order derivatives and integrals is a current research issue. Several numerical methods were proposed in the last decade.

In 1996, Shokooh and Suarez adapted central differences and average acceleration methods for fractional-order damped systems which have one or more degrees of freedom, and applied for systems include 0.5^{th}-order derivatives [17]. In Podlubny's method described in [18] the derivative and integral operators are combined by using matrices and generalized for fractional-orders in order to apply to any linear system easily. If the initial conditions are different from zero, the algorithm needs to be changed by hand. Furthermore, this method can be used to simulate nonlinear fractional-order systems. In 2007, Gejji and Jafari proposed a numerical method in their study [19] that uses Adomian Decomposition method for solving multi-order fractional differential equations that consist of Caputo fractional differential equations.

In this study, a method for finding a numerical solution of fractional-order differential equations was needed and the numerical method proposed by Dorčák is used to simulate the fractional-order systems and fractional-order controllers [20], because the results obtained by this method show a good compatibility between analytical and numerical calculations of the unit-step response of any system, especially with small time steps. Dorčák established a numerical model in his study for the simulation of the dynamic characteristics of fractional-order control systems and analyzed dynamic properties of systems in time-domain.

For a fractional order system defined by the transfer function

$$G(s) = \frac{1}{a_2 s^{\alpha} + a_1 s^{\beta} + a_0} \tag{2.63}$$

which corresponds to the fractional-order differential equation

$$a_2 y^{(\alpha)}(t) + a_1 y^{(\beta)}(t) + a_0 y(t) = u(t) \tag{2.64}$$

where $u(t)$ is the control signal, with initial conditions $y^{(\beta)}(0) = 0$ and $y(0) = 0$, the following relation with "short memory" principle is employed for the approximation of

the fractional derivatives in Equation (2.64):

$$y^{(\alpha)}(t) \approx_{t-L} D_t^{\alpha} y(t) = h^{-\alpha} \sum_{j=0}^{N(t)} b_j y(t - jh), \tag{2.65}$$

where L is memory length and h is time step,

$$N(t) = min\{\left[\frac{t}{h}\right], \left[\frac{L}{h}\right]\} \tag{2.66}$$

The coefficient b_j is calculated with the following recurrent relation:

$$b_0 = 1, \quad b_j = (1 - \frac{1+\alpha}{j}) b_{j-1} \tag{2.67}$$

By using the approximation in Equation (2.65) the following explicit recurrent relation for the calculation of the values y_m $(m = 2, 3, \ldots)$:

$$y_m = \frac{u_m - a_2 h^{-\alpha} \sum_{j=1}^{m} b_j y_{m-j} - a_1 h^{-\beta} \sum_{j=1}^{m} c_j y_{m-j}}{a_2 h^{-\alpha} b_0 + a_1 h^{-\beta} c_0 + a_0} \tag{2.68}$$

with $y_0 = 0$, $y_1 = 0$, $u_0 = 0$, and $u_m = 1$ for $m = 1, 2, \ldots$.

2.6. Geometric and Physical Interpretation of Fractional-Order Integration and Differentiation

Geometric and physical interpretation of integer-order integral and derivative is clear. Therefore they can be simply used in many areas of science. Since fractional-order integral and derivative include n^{th}-order derivatives and $n-folded$ integrals (n is and integer number) and they are generalization of integer-order integral and derivative operations, physical and geometrical interpretations of fractional-order operators can provide a connection to interpretations of integer-order operators. Podlubny has suggested a solution to the problem of geometric and physical interpretation of fractional-

order integration and fractional-order differentiation in [21]. Podlubny introduced a geometric interpretation for the left-sided and right-sided Riemann-Liouville fractional integration, the Riesz potential, and the Feller potential. Based on this, he suggested a physical interpretation of the Riemann-Liouville fractional integration in terms of inhomogeneous and changing time scale. He also suggested physical interpretation for Riemann-Liouville fractional differentiation and for the Caputo fractional differentiation.

2.6.1. Geometric Interpretation of Fractional-Order Riemann-Liouville Integration

Left-sided Riemann-Liouville integral of order α is:

$$ {}_0J_t^{\alpha} f(t) = \frac{1}{\Gamma(\alpha)} \int_0^t f(\tau)(t-\tau)^{\alpha-1} d\tau, \;\; t \geq 0, \tag{2.69} $$

If it is written in the form

$$ {}_0J_t^{\alpha} f(t) = \int_0^t f(\tau) dg_t(\tau), \tag{2.70} $$

$$ g_t(\tau) = \frac{1}{\Gamma(\alpha+1)} \{t^{\alpha} - (t-\tau)^{\alpha}\} \tag{2.71} $$

If the axes are taken as τ, g, and f, and the function $g_t(\tau)$ is plotted for $0 \leq \tau \leq t$ in the plane (τ, g), then along the obtained curve a fence with varying height $f(\tau)$ is formed. So, the top edge of the fence is a three-dimensional line $(\tau, g_t(\tau), f(\tau))$, $0 \leq \tau \leq t$. This fence can be projected onto two surfaces as in Figure 2.2:

- The area of the projection of this fence onto the plane (τ, f) corresponds to the value of the integral

$$ {}_0J_t^{1} f(t) = \int_0^t f(\tau) d\tau; \;\; t \geq 0 \tag{2.72} $$

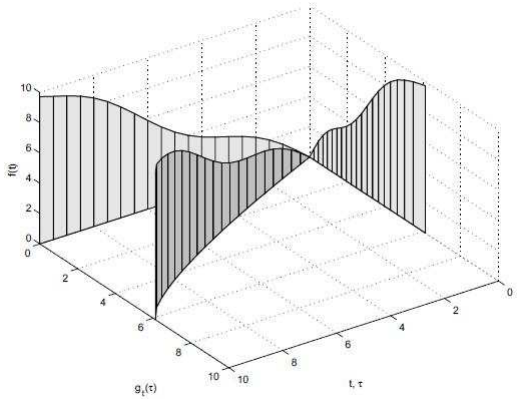

Figure 2.2. The fence and its shadows: ${}_0J_t^1 f(t)$ and ${}_0J_t^\alpha f(t)$ for $\alpha = 0.75,\ f(t) = t + 0.5\sin(t), 0 \le t \le 10$ [21]

- The area of the projection of the same fence onto the plane (g, f) corresponds to the value of the integral Equation (2.71).

While the shadow on the wall (τ, f) is a standard geometric interpretation of the integral, the shadow on the wall (g, f) is a geometric interpretation of the fractional integral for a fixed t.

For the right-sided Riemann-Liouville fractional integral in the form

$$ {}_tJ_0^\alpha f(t) = \int_t^b f(\tau) dh_t(\tau), h_t(\tau) = \frac{1}{\Gamma(\alpha+1)}\{t^\alpha + (t-\tau)^\alpha\} \tag{2.73} $$

there is not any fixed point in the fence base moves alon the line $\tau = b$ in the plane (τ, g) when the fence changes its shape.

All other parts of the geometric interpretation is the same. Both left-sided Riemann-Liouville and right-sided Riemann-Liouville fractional integration includes the classical integration as a particular case.

2.6.2. Physical Interpretation of Fractional-Order Riemann-Liouville Integration

The left-sided Riemann-Liouville fractional integral is given as

$$S_O(t) = \int_0^t \upsilon(\tau) dg_t(\tau) =_0 J_t^\alpha \upsilon(t), \tag{2.74}$$

where $g_t(\tau)$ is given by Equation (2.71).

The fractional integral $S_O(t)$ of the function $\upsilon(\tau)$ can be interpreted as the real distance passed by a moving object for which the local values of its speed $\upsilon(\tau)$ and the local values of its time τ have been recorded. The relationship between τ (which is considered as flowing equably) and the cosmic time (which flows non-equably) is given by the function $g_t(\tau)$ which describes the inhomogeneous time scale that depends on both τ and t.

The left-sided Riemann-Liouville fractional integral of the individual speed $\upsilon\tau$ of a moving object, for which the relationship between its individual time τ and cosmic time T at each individual time instance t is given by the known function $T = g_t(\tau)$, represents the real distance $S_O(t)$ passed by that object.

2.6.3. Physical Interpretation of Fractional-Order Riemann-Liouville Derivative

Left-sided Riemann-Liouville derivative of $S_O(t)$:

$$\upsilon(t) =_0 D_t^\alpha S_O(t) \tag{2.75}$$

The left-sided Riemann-Liouville fractional derivative of the real distance $S_O(t)$ passed by a moving object is equal to the individual speed $\upsilon(\tau)$ of that object.

If the relationship in (2.74) is differentiated with respect to the cosmic time variable t, the following relationship is obtained:

$$v_O(t) = \frac{d}{dt}\,{}_0J_t^{\alpha} v(t) =_0 D_t^{1-\alpha} v(t) \tag{2.76}$$

Therefore, the $(1-\alpha)^{th}$-order Riemann-Liouville derivative of the individual velocity $v(t)$ is equal to the velocity $v_O(t)$ from the viewpoint of the independent observer. For $\alpha = 1$, both velocities are equal: $v_O(t) = v(t)$.

2.6.4. Physical Interpretation of Fractional-Order Caputo Derivative

After applying fractional integration of order $\beta = 1-\alpha$ to both parts of (2.76) the following relationship is obtained:

$$v(t) =_0 J_t^{1-\alpha} v_O(t) =_0 J_t^{1-\alpha} S_O'(t) \tag{2.77}$$

The Caputo fractional derivative has the same physical interpretation as the Riemann-Liouville fractional derivative. If $f(0) = 0$i then the Riemann-Liouville derivative and Caputo derivative of order α $(0 < \alpha < 1)$, are equal.

3. STABILITY ANALYSIS OF FRACTIONAL-ORDER LINEAR SYSTEMS

Stability analysis of classical linear time-invariant (LTI) systems can easily be performed by using Routh-Hurwitz criterion. Stability analysis of fractional-order systems requires visualization of multiple Riemann sheets and conformal map from s to another complex plane w. System dynamics are described by the pole location of the transfer function in w-domain.

3.1. Riemann Sheets

In order to understand the dynamics of any particular system, the nature of the singularities are considered in s-domain, but fractional-order systems that have fractional order transfer functions do not have any poles on the primary Riemann-sheet of the s-plane.

In mathematics, particularly in complex analysis, a Riemann surface is a one-dimensional complex manifold. The main advantage of Riemann surfaces is that multi-valued functions can be defined on a Riemann surface which consist of finite number of Riemann sheets. These sheets are separated by brunch cuts (BC) and each sheet has only one edge at a BC [22]. Brunch cuts are the boundaries of discontinuity for the multi-valued function. In order to get to a secondary Riemann-sheet, it is necessary to go through a branch cut on the primary Riemann-sheet. These branch cuts can also be considered to be singularities on the primary Riemann-sheet of the s-domain too, but the transfer functions in the Laplace domain do not go to infinity at those points [23]. Figure 3.1 shows the Riemann surface for $w = s^{1/3}$.

3.2. $s \rightarrow w$ Plane Mapping for Stability Analysis

If a fractional-order transfer function $G(s) = \frac{1}{s^{\alpha}+1}, (0 < \alpha < 1)$ is considered as an example from [8], then it can be seen that this system do not have any singularity

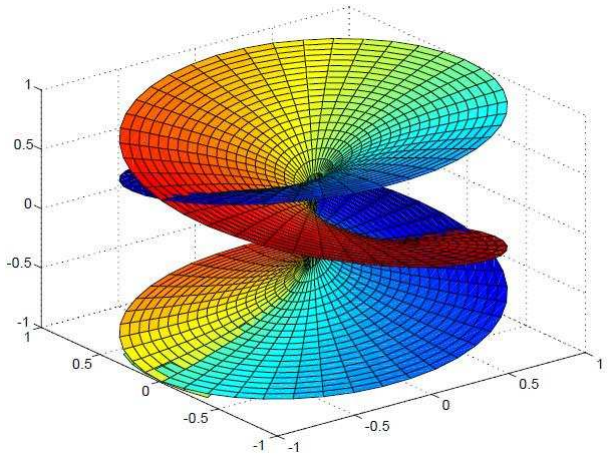

Figure 3.1. Riemann surface for $w = s^{1/3}$ [22]

point (pole) anywhere in the s-plane (Primary Riemann-sheet), but after going through the branch cut on the primary Riemann-sheet, secondary Riemann-sheet will contain the singularity point (pole).

For a fractional-order transfer function, e.g. $G(s) = \frac{b}{s^q+a}$, rather than dealing with fractional powers of s, the analysis is carried on w- plane where $w = s^q$. The mapping from $s \rightarrow w$ is as follows:

$$w = \rho e^{j\phi} = \alpha + j\beta \;\; and \;\; s = re^{j\theta} \tag{3.1}$$

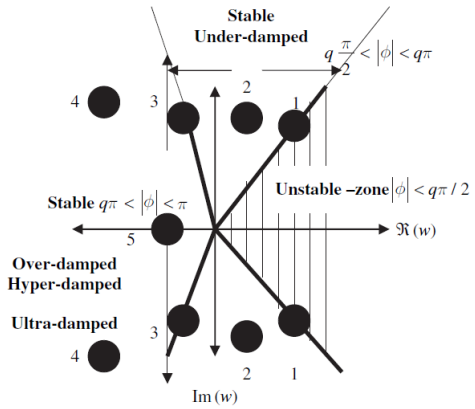

Figure 3.2. w-plane Stability Zones and Various Pole Locations [8]

Defining $w = s^q = (re^{j\theta})^q = r^q e^{jq\theta} = \rho e^{j\phi}$ implies that $\rho = r^q$ and $\phi = q\theta$. With those equalities it is possible to map s-plane into w-plane. The stability boundary becomes $\phi = \pm q\Pi/2$ in the w-plane, whereas $\phi = \pm\Pi/2$ in the s-plane. Then the Right Half Plane (RHP) maps into a "wedge" in the w-plane.

For $q = 1/2$, Figure 3.2 shows the stability zones of w-plane and the properties of the system's response according to the location of the pole. The entire primary sheet of s-plane maps into a w-plane wedge of angle less than $\pm 180q$ degree, while the secondary sheet maps into the remainder of the w-plane.

4. $PI^{\lambda}D^{\mu}$ CONTROLLER

The simplest form of the fractional-order PID controller is the $PI^{\lambda}D^{\mu}$ controller [2] which involves an integrator of order λ and differentiator of order μ, where λ and μ are parameters that can take on real values. The transfer function of such a controller has the following form:

$$G_c(s) = K_p + \frac{K_i}{s^{\lambda}} + K_d s^{\mu} \qquad (4.1)$$
$$(\lambda, \mu > 0)$$

Fractional-order PID controller differs from classical PID controller in terms of the orders of the integral and derivative actions. These orders can take on real values and number of parameters increase. This increase in the number of parameters provides extra degrees of freedom for achieving the design specifications.

It is obvious that classical PID controller can be considered as a special type of $PI^{\lambda}D^{\mu}$ controller with $\lambda = \mu = 1$. It can be expected that $PI^{\lambda}D^{\mu}$ controllers may enhance the systems control performance by creating two new degrees of freedom. Podlubny has shown in [2] the $PI^{\lambda}D^{\mu}$ controller has also the advantage of being less sensitive to changes in the parameters of controlled system. Fractional-order integration and fractional-order differentiation add flexibility to the controller design and allow us to control the real world processes more accurately. The unity feedback control system with $PI^{\lambda}D^{\mu}$ controller shown in figure 4.1 is considered in this study.

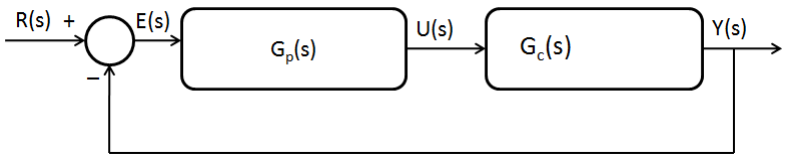

Figure 4.1. Closed-Loop System

The control signal $u(t)$ can be defined with the following fractional-order differential equation:

$$u(t) = K_p e(t) + K_i D^{-\lambda} e(t) + K_d D^{\mu} e(t) \tag{4.2}$$

Fractional-order PID controllers ($PI^{\lambda}D^{\mu}$) allow use of integral and derivative operators of arbitrary order. Furthermore, derivative and integral orders do not need to take on the same values. Yet the advantages of fractional-order PID control are accompanied by some disadvantages. Fractional differential equations accumulate the entire information of the function, thus the computations requires large memory and computation time.

5. TUNING OF INTEGER AND FRACTIONAL-ORDER PID CONTROLLERS

Before designing a controller, it is important to understand the primary objective of control. A common type of control objective is to track a setpoint or a step input. Time-domain performance specifications on setpoint tracking such as rise time, settling time, maximum overshoot, and steady-state error, need to be incorporated into the design of the controller.

In this chapter commonly used PID and $PI^{\lambda}D^{\mu}$ controllers design techniques are described.

5.1. PID Tuning

PID-control is the one of the oldest control approaches and the one most commonly used in industrial applications. It is based on the idea of applying a control input u which is computed as a weighted sum of the tracking error, its first integral, and first derivative within the basic closed-loop control scheme as in Figure 4.1. The design of such a PID controller, usually referred to as PID tuning, consists of choosing the weighting coefficients K_P, K_I, K_D of the proportional, integral, and derivative actions appropriately for a given system (shown by $G_P(s)$ in Figure 4.1) and a set of design specifications. The design specifications are typically given either in frequency domain (gain margin, phase margin, crossover frequency, etc.) or time domain. Time domain specifications refer to the system response to a reference input, which is most commonly taken as a unit-step input. Typical time-domain performance specifications defined on basis of unit-step response are rise-time (t_r), settling-time (t_s) (generally for settling the 2% or 5% tolerance zone around the steady-state value), maximum percentage overshoot (M_p), and steady-state error (ε_{ss}).

The classical PID controller is implemented as follows:

$$G_c(s) = K_P + \frac{K_I}{s} + K_D s \tag{5.1}$$

where K_P is the proportional gain, K_I is the integral gain, and K_D is the derivative gain. A set of monotonic relations given in the literature between time-domain characteristics and PID coefficients valid for an integer-order system controlled by an integer-order PID controller is summarized in Table 5.1.

Table 5.1. Effects of increasing one of the PID coefficients on the closed-loop unit-step response of an integer order linear system

Parameter	Rise-time	Overshoot	Settling-time	Steady-state Error
K_P	Decrease	Increase	Small change	Decrease
K_I	Decrease	Increase	Increase	Eliminate
K_D	Small change	Decrease	Decrease	Small change

Tuning methods for the classical (integer-order) PID controller has a long history. In [24] and in [25] the most common PID tuning techniques are summarized. The Ziegler-Nichols Step Response Method, The Ziegler-Nichols Frequency Response Method, and Dominant Pole Design: The Cohen-Coon Method are the most common PID tuning methods known in the literature. All of these methods belong to the optimization approach based on exploiting systematic knowledge about the system.

5.1.1. The Ziegler-Nichols Step Response Method

Ziegler-Nichols Step Response Method, one of the most popular techniques, is an experimental technique that can only be applicable to open-loop stable plants. This method first characterizes the system with two parameters T and L time constant and delay-time, respectively, obtained from its step response. Graphically, first the point on the step response that has the maximum slope is determined and the tangent is drawn. K is the intersection of the tangent with the vertical axis, while the intersection of the tangent with the horizontal axis gives L. T is the difference between the time that the

system response is K and L. After obtaining T and L, PID controller parameters are determined by the following formulas in terms of T and L:

Table 5.2. PID Parameters Obtained by Ziegler-Nichols Step Response Method

Type of Controller	K_P	K_I	K_D
P	$\frac{T}{L}$	0	0
PI	$0.9\frac{T}{L}$	$\frac{0.27T}{L^2}$	0
PID	$1.2\frac{T}{L}$	$\frac{0.6T}{L^2}$	$0.6T$

System response that obtained by PID controller designed by this method gives a small settling-time and overshoot decays $\frac{1}{4}^{th}$ of its original value.

5.1.2. The Ziegler-Nichols Frequency Response Method

Ziegler-Nichols Frequency Response Method, is a closed-loop tuning method. This method first determines the point where the Nyquist curve of the system intersects the negative real axis. The Ziegler-Nichols frequency response method gives the following formulas for setting the PID controller parameters in terms of critical gain and critical period:

Table 5.3. PID Parameters Obtained by Ziegler-Nichols Frequency Response Method

Type of Controller	K_P	K_I	K_D
P	$0.5K_{cr}$	0	0
PI	$0.45K_{cr}$	$\frac{0.54K_{cr}}{T_{cr}}$	0
PID	$0.6K_{cr}$	$\frac{1.2K_{cr}}{T_{cr}}$	$0.075K_{cr}T_{cr}$

where K_{cr} is the critical gain that oscillation is observed for the first time and T_{cr} is the critical period of oscillation. Since this procedure measures the ultimate gain and ultimate frequency when the system is close to instability, this method needs to be executed carefully in order to avoid damaging the physical system.

Both Ziegler-Nichols methods are widely used to tune PID controller parameters

where the plant dynamics are not precisely known, and they offers simple formulas for determining PID controller parameters. Besides being model-free methods, Ziegler-Nichols have disadvantages too. Since those methods give little emphasis to measurement noise, sensitivity to process variations, the resulting closed-loop system can easily go to instability.

5.1.3. Dominant Pole Design:The Cohen-Coon Method

This technique attempts to locate some poles in order to achieve design specifications [24]. Those poles are three dominant poles: A pair of complex poles and a real pole. This method is based on the first-order plant model with deadtime:

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

PID controller parameters are obtained in terms of k, T, and L as:

$$K_P = \frac{1.35(1-0.82b)}{a(1-b)}$$
$$K_I = \frac{1.35(1-0.82b)(1-0.39b)}{aL(1-b)(2.5-2b)}$$
$$K_D = \frac{1.35L(0.37-0.37b)}{a(1-b)}$$

where

$$a = \frac{kL}{T}$$
$$b = \frac{L}{L+T}$$

5.2. $PI^{\lambda}D^{\mu}$ Tuning

The fundamental difficulty in tuning λ and μ is due to the fact that for systems of arbitrary order there exists no monotonous relationship between these parameters and

time-domain performance criteria like overshoot (M_p), rise-time (t_r), and settling-time (t_s). Furthermore, as shown in [4] by qualitative analysis and simulations, the closed-loop behavior of the system exhibits very sensitive dependence on μ in particular, which implies that objective functions will not possess sufficient smoothness (at least among the μ axis) necessary for gradient-based optimization methods. This fact suggests that the tuning of λ and μ would lend itself better either to model-based optimization methods (performed on the known model of the specific system at hand) or to stochastic (blind search) optimization techniques (applicable for the control of linear systems without knowing the mathematical system model). As far as I have been able to observe, all fractional-order PID tuning methods given in the literature are model-based methods: Some of them extend the classical frequency-domain techniques to fractional-order [26] and [27], some of them optimize pole-placement for a given system by the PSO method [7].

There are different methods for tuning $PI^{\lambda}D^{\mu}$ parameters. The $PI^{\lambda}D^{\mu}$ parameters are calculated for a fractional-order integrator of order α in [28] by a model-based method based on the parameters α and w_c which are the order and the gain cross-over frequency, respectively. In [4] fractional-order PID controller is designed by a model-based method according to desired gain margin and phase margin in order to meet the stability robustness of the feedback control loop. Different design specifications have been used to tune $PI^{\lambda}D^{\mu}$ controller parameters such as maximum overshoot and rise time as in [7] by replacing the poles as to satisfy the requirements. There are different $PI^{\lambda}D^{\mu}$ tuning algorithms in the literature as in [27], [30], [31], and [32].

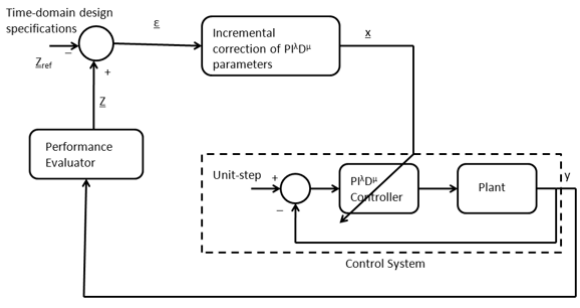

Figure 5.1. Model-free $PI^{\lambda}D^{\mu}$ tuning scheme

The aim of this study is the design of a model-free algorithm which can tune the five parameters of fractional-order PID controller parameters (K_P, K_I, K_D, λ, μ) on basis of the evaluation of the unit-step response obtained from the system. The $PI^{\lambda}D^{\mu}$ tuner scheme for this study is given in Figure 5.1.

In the next chapter, Particle Swarm Optimization technique and its improved version, namely Knowledge-Supported PSO Algorithm, will be described in detail.

6. PARTICLE SWARM OPTIMIZATION AND ITS IMPROVEMENT

6.1. Particle Swarm Optimization Technique

Many popular optimization algortihms are deterministic, like the gradient-based algorithms. Particle Swarm Optimization (PSO) algorithm belongs to the Evolutionary Algorithm family and is a stochastic algorithm that does not need the gradient information. This property allows to use PSO on functions where the gradient is either unavailable or computationally expensive to obtain.

The PSO algorithm [5] is a population based stochastic optimization technique which attempts to mimic the natural process of group communication of individual knowledge. PSO is an evolutionary computation technique and has many similarities with Genetic Algorithms (GA). Compared to GA, PSO is easier to implement and it doesn't work only with binary numbers.

The *swarm* is composed of particles and each particle has position and velocity. The algorithm searches the solution space by swarming the particles. Each particle updates its position and velocity according to its best solution and the best solution of the other particles in the past. All particles have fitness values which are evaluated by the fitness function to be optimized.

Let the swarm consists of N particles moving around a D-dimensional search space. The i^{th} particle is denoted by $X_i = (x_{i1}, x_{i2}, \cdots, x_{iD})$ and the current velocity is $V_i = (v_{i1}, v_{i2}, \cdots, v_{iD})$. Its best previous solution (pbest) is $P_i = (p_{i1}, p_{i2}, \cdots, p_{iD})$ and the best solution obtained so far by the whole swarm (gbest) is $P_g = (p_{g1}, p_{g2}, \cdots, p_{gD})$.

Each particle updates its position and velocity with the following equations and

they move to pbest and gbest locations at each iteration:

$$v_{iD}(k+1) = w_k v_{iD}(k) + c_1\varphi_1(p_{iD}(k) - x_{iD}(k)) + c_2\varphi_2(p_{gD}(k) - x_{iD}(k)) \quad (6.1)$$

$$x_{iD}(k+1) = x_{iD}(k) + v_{iD}(k+1) \quad (6.2)$$

Here c_1 and c_2 are positive constants, called cognitive learning rate and social learning rate, respectively. φ_1 and φ_2 are two random functions in the range [0,1]. w is the inertia factor and it decreases from 0.9 to 0.4 during to optimization process with the equation:

$$w_k = w_{max} - (\frac{w_{max} - w_{min}}{k_{max}})k \quad (6.3)$$

6.2. Knowledge-Supported PSO

Since PSO is based on particles' personal and social experiences, it is suitable for blind search, but normally living beings are also endowed with a priory knowledge represented in compact form. In simple organisations such knowledge is coded in the form of reflex which provide crucial advantages to these living beings. Basic PSO algorithm does not allow us to use this knowledge.

The Knowledge-Supported PSO algorithm introduces a structure, which allows for each particle a transition between different modes of operation; namely particles can operate within a sociocognitive mode as in the basic PSO algorithm but also can switch to a mode where they employ some systematic knowledge in a "reflex-like" manner. Systematic knowledge is integrated to PSO algorithm by Reflex blocks.

The scheme in Figure 6.1 will be referred to as "Knowledge-Guided PSO" algorithm. For a given optimisation problem Reflex blocks need to be formulated according to available systematic knowledge.

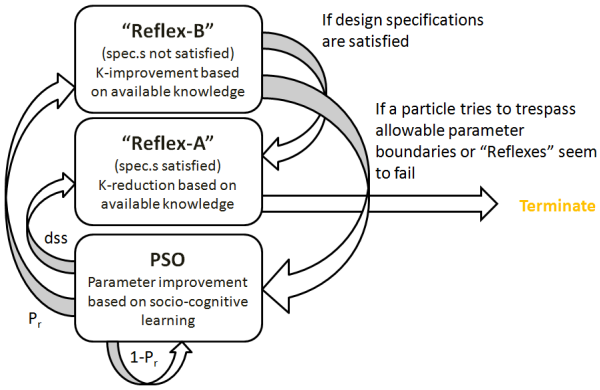

Figure 6.1. Switching diagram between modes for each particle

6.3. $PI^{\lambda}D^{\mu}$ Tuning with Knowledge-Supported PSO

The aim of this thesis was the design of a model-free algorithm, which can tune the five parameters $(K_P, K_I, K_D, \lambda, \mu)$ on basis of an evaluation of the unit step response obtained from the system. The PSO algorithm performs well, but while some of the effects of K_P, K_I, and K_D are known, the effects of λ and μ parameters are unknown on the system response. By combining the general information about integer-order PID controller parameters' effects in the literature (Table 5.1) and the information that we have gained by the experiments on different types of systems, the systematic interaction given as in Table 6.1.

Table 6.1. Effects of increasing one of the K_P, K_I, K_D coefficients of a $PI^{\lambda}D\mu$ controller on the closed-loop unit-step response of an arbitrary linear system

Parameter	Overshoot	Rise-time	Settling-time	Steady-state Error
K_P	-	Decrease	-	-
K_I	Increase	Decrease	-	Eliminate
K_D	Decrease	-	-	-

In this study, the design specifications are predefined overshoot (M_p), rise-time (t_r), and settling-time (t_s) values and also steady-state error is checked at every step of the algorithm. The design method tunes the $PI^{\lambda}D^{\mu}$ controller parameters to ensure the design specifications without requiring the mathematical model of system. System

Table 6.2. Initial values of each particle

particle ♯	λ	μ	K_P	K_I	K_D
1	0	0	$random$	$\emptyset$	$\emptyset$
2	0	1	$random$	$\emptyset$	$random$
3	0	2	$random$	$\emptyset$	$random$
4	1	0	$random$	$random$	$\emptyset$
5	1	1	$random$	$random$	$random$
6	1	2	$random$	$random$	$random$
7	2	0	$random$	$random$	$\emptyset$
8	2	1	$random$	$random$	$random$
9	2	2	$random$	$random$	$random$

stability is anyway guaranteed if the design specifications are satisfied. Systematic knowledge used in the reflex modes is based on the effects of increasing the K_P, K_I, and K_D coefficients on the system response.

Knowledge-supported PSO algorithm for tuning fractional-order PID controller gives priority to integer-order PID controller different from the several methods for tuning fractional-order PID controllers in the literature by initializing the particles with the combinations of derivatives and integrals (μ and λ) of order 0, 1, and 2. The reason that λ and μ are initialized with integer numbers is the ease of implementation of integer-order PID controller. K_P, K_I, and K_D values are initialized randomly. Also, this method does not need system's mathematical model to tune the controller's parameters. Table 6.2 shows the initial values of each particles:

This type of initialization as in Table 6.2 and $Reflex - B$ mode, namely K-tuning which will be described below, provides this algorithm to find integer-order PID controllers for a system if there exists.

This method established as including two modes, namely Reflex parts where only K_P, K_I, and K_D parameters of the fractional-order controller are tuned and socio-

cognitive part where the original the PSO velocity update formula is used.

All particles do not need to be at one mode at the same time. Each particle's velocity and position are updated according to the mode's self update equation. The fitness function for this algorithm is:

$$f = \varepsilon_{M_p}^2 + \varepsilon_{t_r}^2 + \varepsilon_{t_s}^2 + \varepsilon_{ss}^2 \tag{6.4}$$

where $\varepsilon_{M_p}, \varepsilon_{t_r}, \varepsilon_{t_s}, \varepsilon_{ss}$ are the percentage errors between the reference values of design specifications and obtained values of M_p, t_r, t_s, and steady state value.

All of the particles initially start from mode $Reflex-B$ where only proportional, integral, and derivative gains are tuned. When the particles are in $Reflex-B$ mode, a matrix that determines the increment or decrement amount of the PID parameters by multiplying with the percentage error of each particle. This matrix's elements are determined according to the Table 6.1 and this matrix is used for calculating the amount of changes in K-parameters, i.e., ΔK_P, ΔK_I, and ΔK_D values:

$$v_{iD}(k+1) = \begin{pmatrix} 0 & \alpha & 0 & 0 \\ -\gamma & \gamma & 0 & 0 \\ \beta & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \varepsilon_{M_p} \\ \varepsilon_{t_r} \\ \varepsilon_{t_s} \\ \varepsilon_{ss} \end{pmatrix} = \begin{pmatrix} \Delta K_P \\ \Delta K_I \\ \Delta K_D \\ 0 \\ 0 \end{pmatrix} \tag{6.5}$$

where α, γ, and β is chosen as $K_{P_{max}}/30$, $K_{I_{max}}/10$, and $K_{D_{max}}/10$, respectively. Update formula used in Reflex block gives priority to the elimination of steady-state error.

Therefore, if the steady-state error is differen from zero, then the matrix becomes:

$$\begin{pmatrix} 0 & \alpha & 0 & 0 \\ -\gamma & \gamma & 0 & \gamma \\ \beta & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{6.6}$$

The position update formula at k^{th} iteration for i^{th} particle is defined as:

$$x_{iD}(k+1) = x_{iD}(k) + v_{iD}(k+1) \tag{6.7}$$

If a particle is equal to its previous value, which means the borderline case, or its fitness value is larger than its previous fitness value, then this particle is moved to $Socio-cognitive$ mode to update both K_P, K_I, K_D and λ, μ values. In this mode, the velocity update formula is as follows:

$$v_{iD}(k+1) = c_1\varphi_1(p_{iD}(k) - x_{iD}(k)) + c_2\varphi_2(p_{gD}(k) - x_{iD}(k))$$

After calculating the new position of the particle in the $socio-cognitive$ mode, then according to the assigned probability of switching to the $Reflex-B$ mode, it is determined that the particle moves to which mode. The particle moves to the $Reflex-B$ mode with probability P_{reflex} or moves to the $socio-cognitive$ mode with probability $1-P_{reflex}$.

If one of the particles which gives the global best fitness function value- namely the global best particle- satisfies the design specifications, then the particle moves to the $Reflex-A$ mode where smaller K_P, K_I, K_D parameters that still satisfies the

specifications are searched. The decrement amounts of the parameters are:

$$\begin{aligned} \Delta K_P &= \alpha \varepsilon_{t_r} \\ \Delta K_I &= \gamma \varepsilon_{t_r} \\ \Delta K_D &= \beta \varepsilon_{M_p} \end{aligned} \tag{6.8}$$

7. EXAMPLES AND PERFORMANCE OF THE PROPOSED METHOD

Four different types of systems are used to illustrate the performance of proposed method (Knowledge-Supported PSO algorithm) on the fractional-order PID controller tuning method.

7.1. Example 1

The first system includes multiple fractional-order components in its transfer function. The first system is [33]:

$$G_P(s) = \frac{1}{0.8s^{2.2} + 0.5s^{0.9} + 1} \tag{7.1}$$

Design specifications are:

$$M_p = 3\%, t_r = 0.15s, t_s = 0.25s \tag{7.2}$$

This system corresponds to three-term fractional-order differential equation as follows:

$$0.8y^{(\alpha)}(t) + 0.5y^{(\alpha)} + y(t) = u(t) \tag{7.3}$$

with zero initial-conditions.

Table 7.1 shows the comparison of performances of classical Particle Swarm Optimization (PSO) technique and Knowledge-supported PSO technique. Since particles are initialized randomly for both algorithms, they have been run fifteen times to gain information about their performances. According to the Table 7.1, the proposed method

KS-PSO has less computation time and variance of computation time for its finding a solution that satisfies design specifications.

Table 7.1. Comparison of the performances of PSO and knowledge-supported PSO algorithms in terms of computation time for the system 7.1

Method	Average computation time (seconds)	Variance of computation time (sec^2)
PSO	17.7579	547.7344
Knowledge-Supported PSO	10.6281	23.4866

The computation times in Table 7.1 include the time spent for simulation of the fractional-order system. Simulating the fractional-order system spends the 99.27% for KS-SPO and 99.38% for pure PSO of the total time spent for the algorithm. But in situation where $PI^{\lambda}D^{\mu}$ tuning is performed on an actual system, unit-step responses will be directly obtained from the system, such that computation time will be much shorter. Table 7.2 shows the corrected computation time that spent for system simulation is excluded.

Table 7.2. Corrected computation time excluding the time spent for the system simulation

Method	Average computation time (seconds)	Variance of computation time (sec^2)
PSO	0.1101	0.0211
Knowledge-Supported PSO	0.0776	0.0013

Parameter boundaries are set to $1 \leq K_P \leq 700$, $0 \leq K_I \leq 300$, $0 \leq K_D \leq 300$, $0 \leq \lambda \leq 2$, $0 \leq \mu \leq 2$ and probability of switching to reflex mode is chosen as $P_{reflex} = 0.5$. As an example solution found by Knowledge-supported PSO method the following fractional-order PID controller is given:

$$G_c(s) = 143.6649 + \frac{273.5837}{s^{0.9986}} + 194.6859s^{1.6680} \tag{7.4}$$

With this controller the closed-loop system's step-response characteristicas are obtained as $M_p = 0.45\%$, $t_r = 0s$, $t_s = 0.02s$, and steady-state error is zero. After Reflex-A mode, this fractional-order PID controller's proportional, integral, and derivative coefficients are reduced to:

$$G_c(s) = 73.6649 + \frac{183.5837}{s^{0.9986}} + 121.6837s^{1.6680} \tag{7.5}$$

Time-domain performance criteria becomes:

$$M_p = 0.93\%, t_r = 0s, t_s = 0.045s$$

This result reveals that closed-loop system satisfies the design specifications. The fractional-order PID controller found by PSO algorithm is as follows as:

$$G_c(s) = 2.5537 + \frac{33.7670}{s^{1.4981}} + 252.4824s^{1.0797} \tag{7.6}$$

which makes the closed-loop system results as $M_p = 1.97\%$, $t_r = 0.015s$, $t_s = 0.025s$, and steady-state error is zero.

Figure 7.1, 7.2, 7.3, 7.4 illustrate step response obtained by KS-PSO, fitness function values of KS-PSO algorithm, step-response obtained by PSO, fitness function values of PSO algorithm, respectively.

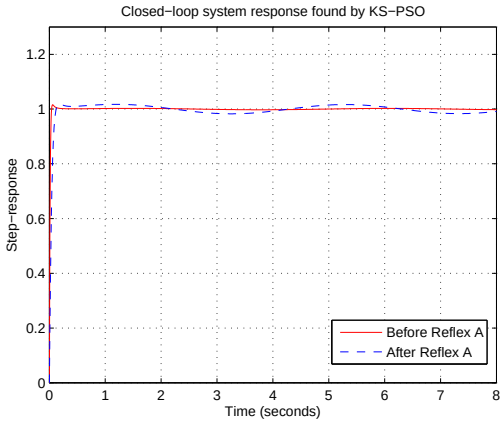

Figure 7.1. Closed-loop system response obtained by KS-PSO for system 7.1

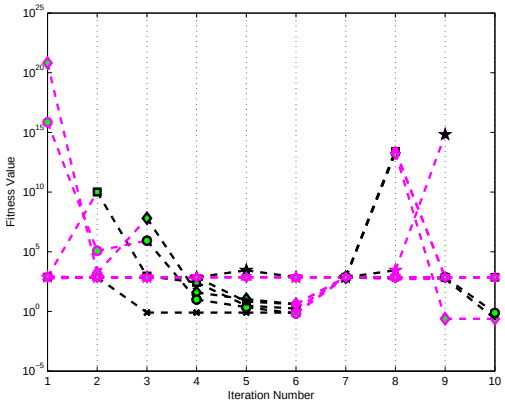

Figure 7.2. Fitness function values calculated by KS-PSO of each particle vs. iteration number system 7.1

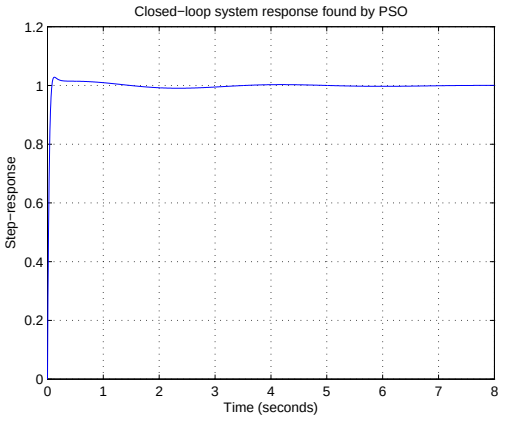

Figure 7.3. Closed-loop system response obtained by PSO for system 7.1

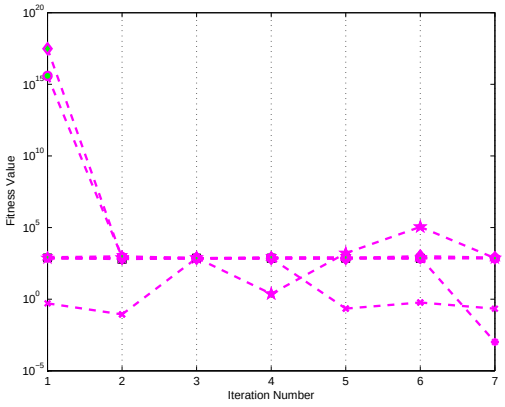

Figure 7.4. Fitness function values calculated by PSO of each particle vs. iteration number for system 7.1

Probability of switching to the Reflex block is determined according to Figure 7.5.

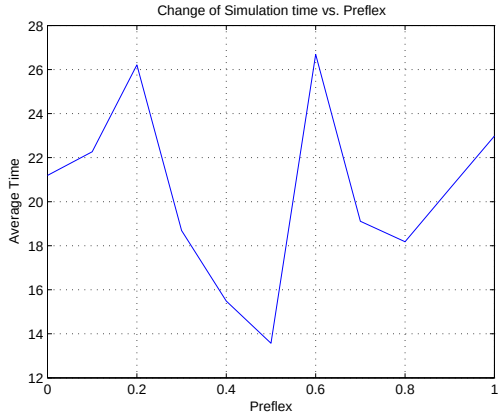

Figure 7.5. Change of average computation time vs. increasing Preflex for system 7.1

7.2. Example 2

The second system is a hypothetical system which is chosen to be an example for a fractional-order system with zero:

$$G_P(s) = \frac{s^{0.3}}{2s^{1.2} + 3.45} \tag{7.7}$$

Design specifications are:

$$M_p = 2\%, t_r = 0.2s, t_s = 0.3s$$

This system has been chosen to illustrate the performance of Knowledge-Supported PSO algorithm's performance on systems having zeros. For this system probability of switching to Reflex block is chosen as $P_{reflex} = 0.6$. Boundary conditions on fractional-order PID controller parameters are $1 \leq K_P \leq 100$, $0 \leq K_I \leq 50$, $0 \leq K_D \leq 50$, $0 \leq \lambda \leq 2$, $0 \leq \mu \leq 2$. The comparison of two methods for the second example is given in Table 7.3.

The corrected computation time excluding time spent for system simulation for system 7.7 is given in Table 7.4

Table 7.3. Comparison of the performances of PSO and knowledge-supported PSO algorithms in terms of computation time for the system 7.7

Method	Average computation time (seconds)	Variance of computation time (sec^2)
PSO	121.1606	2854.1
Knowledge-Supported PSO	16.5817	175.6179

Table 7.4. Corrected computation time excluding the time spent for the system simulation

Method	Average computation time (seconds)	Variance of computation time (sec^2)
PSO	0.3271	0.0208
Knowledge-Supported PSO	0.0448	0.0013

A solution that satisfies the design specifications, found by KS-PSO is given as follows:

$$G_c(s) = 10.7365 + \frac{37.0026}{s^{1.0382}} + 12.3747s^{0.1162} \tag{7.8}$$

The closed-loop system's step response characteristics are obtained as:

$$M_p = 0\%, t_r = 0.105s, t_s = 0.22s \tag{7.9}$$

and steady-state error is zero. This controller obtained before the best particle switching to Reflex A mode, where the smaller PID controller parameters are looked for. At the end of the Refex A mode, the controller that still satisfies the design specifications is obtained as:

$$G_c(s) = 8.6531 + \frac{33.8776}{s^{1.0382}} + 7.9289s^{0.1162} \tag{7.10}$$

With this controller, closed-loop system's step response characteristics are obtained as$M_p = 1.53\%$, $t_r = 0.1s$, $t_s = 0.155s$ and steady-state error is zero.

The PID controller obtained by using pure PSO algorithm which gives the step-response characteristics as $M_p = 1.4\%$, $t_r = 0.165s$, $t_s = 0.285s$ and zero steady state error, is obtained as:

$$G_c(s) = 6.2907 + \frac{33.5193}{s} + 6.3271s^{0.2445} \tag{7.11}$$

Figure 7.6, 7.7, 7.8, 7.9 illustrate step response obtained by KS-PSO, fitness function values of KS-PSO algorithm, step-response obtained by PSO, fitness function values of PSO algorithm, respectively.

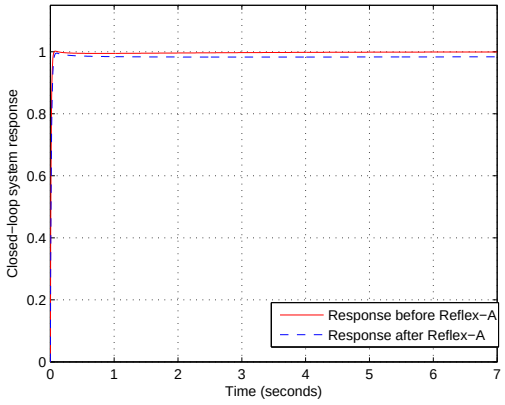

Figure 7.6. Closed-loop system response obtained by KS-PSO for system 7.7

Probability of switching to the Reflex block for system 7.7 is determined according to Figure 7.10.

7.3. Example 3

The third system, includes only one fractional order term in its transfer function, given in the equation below [1]:

$$\frac{1}{39.69s^{1.26} + 0.598} \tag{7.12}$$

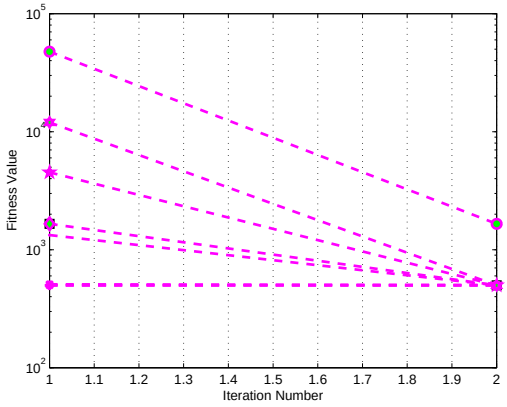

Figure 7.7. Fitness function values calculated by KS-PSO of each particle vs. iteration number system 7.7

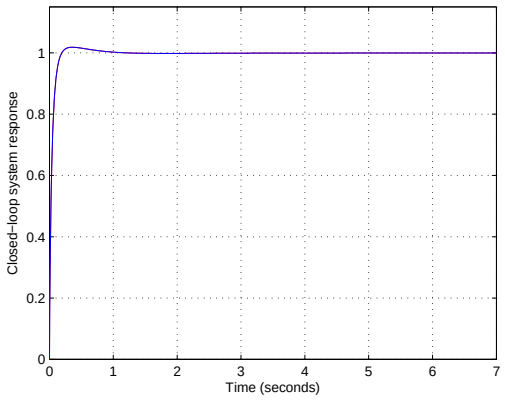

Figure 7.8. Closed-loop system response obtained by PSO for system 7.7

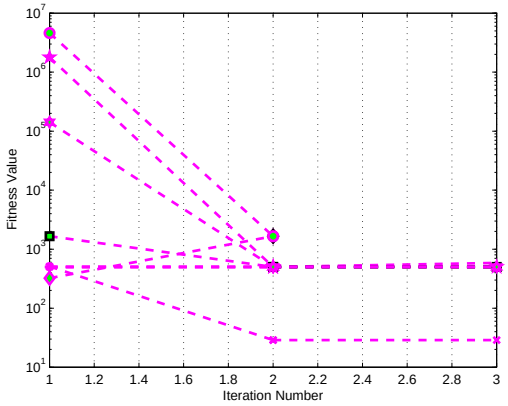

Figure 7.9. Fitness function values calculated by PSO of each particle vs. iteration number for system 7.7

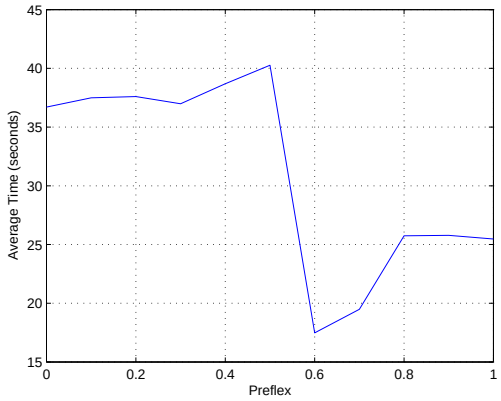

Figure 7.10. Change of average computation time vs. increasing Preflex for system 7.7

Design specifications:

$$M_p = 3\%, t_r = 0.3s, t_s = 0.4s \tag{7.13}$$

This system is chosen to illustrate the superiority of the proposed method on systems that is open-loop unstable and has large steady-state error. Comparison of KS-PSO and pure PSO algorithms for this system is given in table 7.5:

Table 7.5. Comparison of the performances of PSO and knowledge-supported PSO algorithms in terms of computation time for the system 7.12

Method	Average computation time (seconds)	Variance of computation time (sec^2)
PSO	43.4307	2.4793
Knowledge-Supported PSO	33.3075	531.3103

This comparison is not a fair comparison, because the number of finding a solution for fifteen repetitions of the same algorithm is not equal to each other. Table 7.6 shows the comparison of success of the two algorithms.

Knowledge-supported PSO algorithm found the suitable fractional-order system

Table 7.6. Comparison of the successful trials of PSO and knowledge-supported PSO algorithms for fifteen trials system 7.12

Method	Number of trials	Number of trials with success
PSO	15	2
Knowledge-Supported PSO	15	9

for this system as:

$$G_c(s) = 471.0802 + \frac{61.7211}{s^{0.3323}} + 156.7785s^{0.7724} \tag{7.14}$$

which gives $M_p = 1.55\%$, $t_r = 0.205s$, $t_s = 0.38$ and zero steady-state error. After Reflex-A mode proportional, integral, and derivative coefficients are reduced to $K_P = 457.0802$, $K_I = 43.7211$, and $Kd = 129.2916$. Then the controller becomes as follows:

$$G_c(s) = 457.0802 + \frac{43.7211}{s^{0.3323}} + 129.2916s^{0.7724} \tag{7.15}$$

With this parameters, the closed-loop system response characteristics are obtained as $M_p = 1.85\%$, $t_r = 0.22s$, $t_s = 0.38s$.

The fractional-order PID controller parameters calculated by pure PSO algorithm is as follows:

$$G_c(s) = 1.4918 + \frac{150.8317}{s^{0.0265}} + 300s^{0.3371} \tag{7.16}$$

which gives $M_p = 0.5\%$, $t_r = 0.215s$, $t_s = 0.365s$.

Figure 7.11, 7.12, 7.13, 7.14 illustrate step response obtained by KS-PSO, fitness function values of KS-PSO algorithm, step-response obtained by PSO, fitness function values of PSO algorithm, respectively.

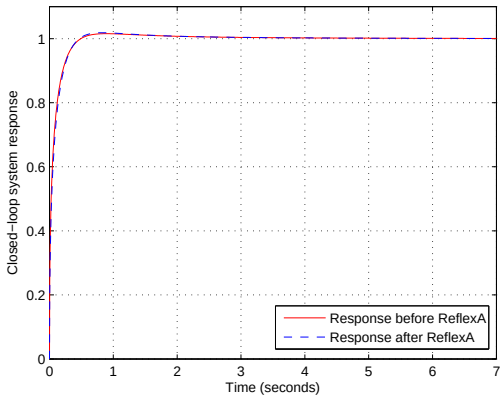

Figure 7.11. Closed-loop system response obtained by KS-PSO for system 7.12

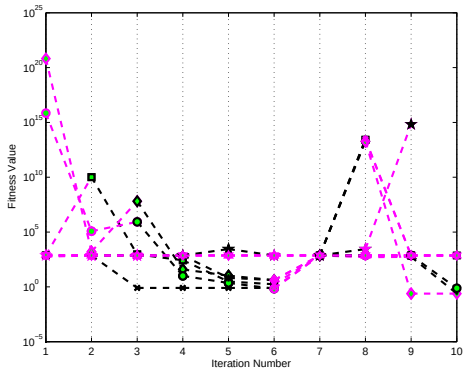

Figure 7.12. Fitness function values calculated by KS-PSO of each particle vs. iteration number system 7.12

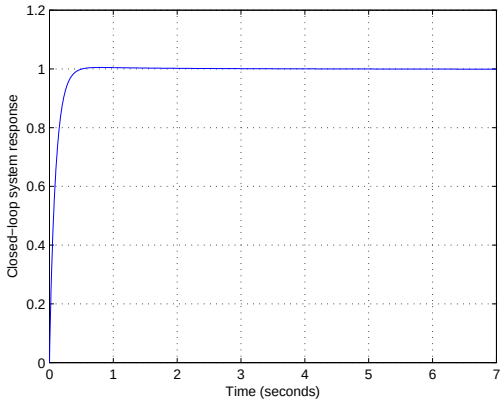

Figure 7.13. Closed-loop system response obtained by PSO for system 7.12

Probability of switching to the Reflex block for system 7.12 is determined according to Figure 7.15.

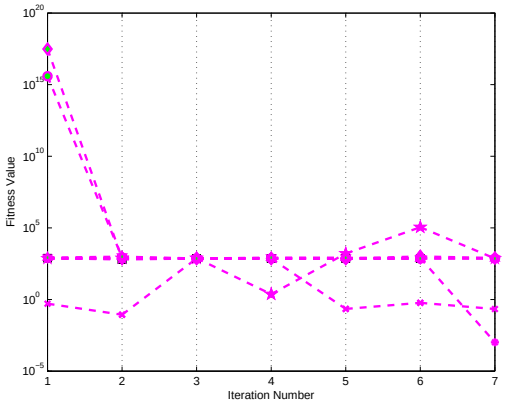

Figure 7.14. Fitness function values calculated by PSO of each particle vs. iteration number for system 7.12

Figure 7.15. Change of average computation time vs. increasing Preflex for system 7.12

7.4. Example 4

In order to show that the fractional-order PID controller and the proposed tuning method is not only applicable for fractional-order systems but also integer-order systems, the following system is chosen [30]:

$$G_P(s) = \frac{0.25}{s(s+1)} \tag{7.17}$$

Design specifications for system 7.17 are:

$$M_p = 1\%, t_r = 0.05s, t_s = 0.1s \tag{7.18}$$

Switching to $Reflex - B$ mode probability is chosen as $P_{reflex} = 0.8$. For this system, boundary conditions are $1 \leq K_P \leq 700$, $0 \leq K_I \leq 200$, $0 \leq K_D \leq 200$, $0 \leq \lambda \leq 2$, $0 \leq \mu \leq 2$.

The comparison of the performances of PSO and KS-PSO in terms of computation time is given in Table 7.7.

Table 7.7. Comparison of the performances of PSO and knowledge-supported PSO algorithms in terms of computation time for the system 7.17

Method	Average computation time (seconds)	Variance of computation time (sec^2)
PSO	34.0505	0
Knowledge-Supported PSO	26.1445	1003.2

For the sake of being a fair comparison, it is suitable to give the comparison in terms of successful trials as in Table 7.8.

Table 7.8. Comparison of the successful trials of PSO and knowledge-supported PSO algorithms for fifteen trials system 7.17

Method	Number of trials	Number of trials with success
PSO	15	1
Knowledge-Supported PSO	15	10

To give an example, knowledge-supported PSO algorithm found an integer-order PD controller while the PSO algorithm found fractional-order PID controller. The controller found by knowledge-supported PSO algorithm is:

$$G_C(s) = 286.3446 + 200s \tag{7.19}$$

With this controller, the closed-loop system characteristics are obtained as $M_p = 0.68\%$,

$t_r = 0.05s$, $t_s = 0.085s$, and the steady-state value is 1. After the calculations done at Reflex-A mode, the parameters are reduced to:

$$G_C(s) = 286.3446 + 180.3391s \tag{7.20}$$

This controller gives time-domain performance criteria as $M_p = 1\%$, $t_r = 0.05s$, $t_s = 0.09s$, and zero steady-state error.

Fractional-order PID controller parameters are calculated by PSO algorithm as follows:

$$G_C(s) = 2.1908 + 200s^{0.8962} \tag{7.21}$$

Closed-loop system response characteristics were obtained as $M_p = 0.14\%$, $t_r = 0.06s$, $t_s = 0.095s$, and the steady-state value is one.

Figure 7.16, 7.17, 7.18, 7.19 illustrate step response obtained by KS-PSO, fitness function values of KS-PSO algorithm, step-response obtained by PSO, fitness function values of PSO algorithm, respectively.

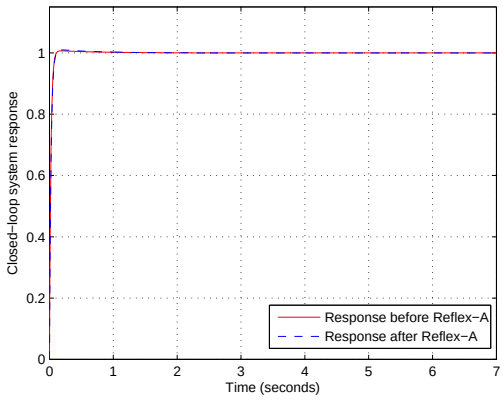

Figure 7.16. Closed-loop system response obtained by KS-PSO for system 7.17

Probability of switching to the Reflex block for system 7.17 is determined according to Figure 7.20 For this example the probability value which gives the minimum fitness value was chosen.

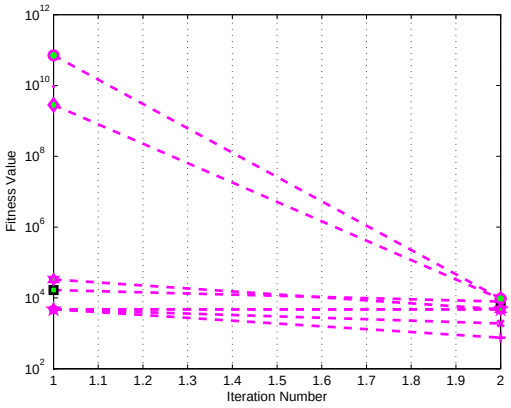

Figure 7.17. Fitness function values calculated by KS-PSO of each particle vs. iteration number system 7.17

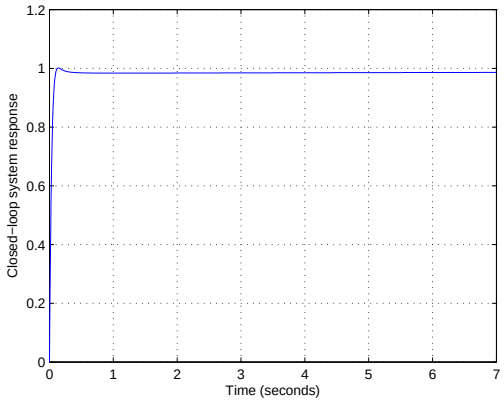

Figure 7.18. Closed-loop system response obtained by PSO for system 7.17

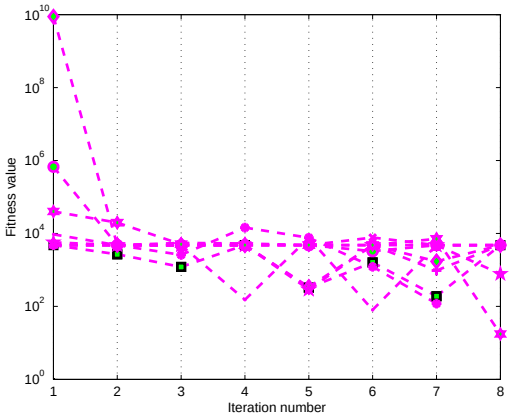

Figure 7.19. Fitness function values calculated by PSO of each particle vs. iteration number for system 7.17

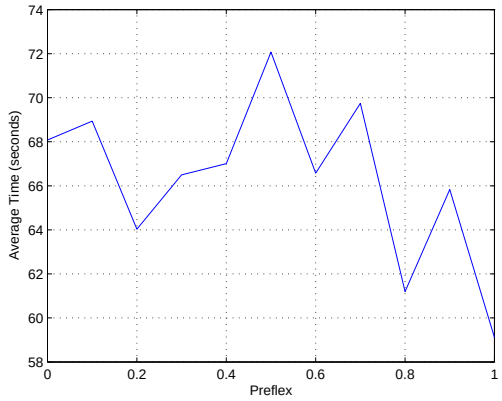

Figure 7.20. Change of average computation time vs. increasing Preflex for system 7.17

Probability of switching from Socio-cognitive learning mode to Reflex-B mode is determined empirically which gives the minimum time that needs the algorithm to find the suitable controller for a given system. Results show that P_{reflex} values between [0.5,1] which increases the switching to Reflex mode probability increases the performance of the algorithm, which reveals that systematic knowledge is effective on finding solutions for an optimisation algorithm which belongs to "blind search category". Using systematic knowledge more frequently decreases the required time for finding the solution.

8. DISCUSSION AND CONCLUSIONS

In this thesis, the task of tuning a fractional-order PID controller has been accomplished using a novel optimisation algorithm which is an extension of the popular stochastic search method known as Particle Swarm Optimisation dubbed as Knowledge-Supported PSO allows the integration of any deterministic relation between the free variables and the objective of optimisation to the stochastic search strategy of the basic PSO algorithm. In optimisation problem where any such deterministic relation is available Knowledge-Supported PSO is expected to have superior performance with respect to pure PSO. Model-free tuning of the parameters of a $PI^{\lambda}D^{\mu}$ controller to be used on an arbitrary linear system is a problem where some effects of some parameters (free variables) on the optimisation objectives (such as M_p, t_r, t_s) are known. A KS-PSO algorithm exploiting this knowledge has been developed for this specific problem. Simulations of $PI^{\lambda}D^{\mu}$ tuning for four different plants with both PSO and KS-PSO have revealed the superiority of the latter in terms of the following criteria:

- Lower average computation time, the number of operations until the optimisation algorithm finds a solution satisfying the design specifications: KS-PSO, whenever it is in the Reflex mode, changes optimisation parameters in a deterministic manner in the correct "improvement" direction and then saves unnecessary operations made for search by trial and error. Actually computation time depends on two factors: The number of computation time per iteration (X) and number of iterations until an acceptable solution is found (M_f). M_f of PSO can be reduced by using more particles, which constitutes the actual strength of PSO. But this would give rise to a proportional increase in X. Most probably for a given problem an optimal number of particles in pure PSO should exist but no a priori knowledge about it is available. The probability P_{reflex} determines the probability of a particle that switches to reflex mode and Figure **??** reveals that, if P_{reflex} is chosen between [0.5,1] which means systematic knowledge mode is used more frequently.
- Lower variance of computation time, the variance of computation time of different

runs of the same optimisaiton problem: Performance tables of the examples reveal that, pure PSO algorithm has higher variance of computation time, which is caused by stochastic computations. With the help of systematic knowledge in KS-PSO algorithm, the time passed until finding a solution does not vary as much as in pure PSO algorithm.

- Ability to find an integer-order solution if possible:
  i) If the algorithm can find parameters as integer-order PID controller, this solution is preferable because it can be more easily implemented.
  ii) This preference is directly integrated into the Reflex mechanism by starting all particles from integer λ and μ values.
  iii) Because pure PSO does not offer any possibility of imposing this preference it can not find any integer-order solution even if it is initialized with integer λ and μ values.

To sum up, Knowledge-Supported PSO performs better than the original PSO algorithm as a framework combining deterministic system specific knowledge with stochastic search. However, it involves an extra effort on behalf of the user who is supposed to develop a "Reflex" block that summarizes his/her a priori knowledge about the optimisation system.

The secondary result that can be obtained from the simulations that, fractional-order PID controller can stabilize any arbitrary linear system more easily than its integer-order counterpart. The fractional-order PID controller achieves better control performance by satisfying the design specifications while the integer-order PID controller can not find a solution that meets the specifications.

APPENDIX A: BASIC FUNCTIONS AND EQUATIONS IN FRACTIONAL CALCULUS

A.1. Mittag-Leffler Function's Identities and Relations to Other functions

$$E_{1,1}(z) = \sum_{k=0}^{\infty} \frac{z^k}{\Gamma(k+1)} = \sum_{k=0}^{\infty} \frac{z^k}{k!} = e^z \tag{A.1}$$

$$E_{1,2}(z) = \sum_{k=0}^{\infty} \frac{z^k}{\Gamma(k+2)} = \sum_{k=0}^{\infty} \frac{z^k}{(k+1)!} = \frac{1}{z}\sum_{k=0}^{\infty} \frac{z^{k+1}}{(k+1)!} = \frac{e^z - 1}{z} \tag{A.2}$$

$$E_{1,3}(z) = \sum_{k=0}^{\infty} \frac{z^k}{\Gamma(k+3)} = \sum_{k=0}^{\infty} \frac{z^k}{(k+2)!} = \frac{1}{z^2}\sum_{k=0}^{\infty} \frac{z^{k+2}}{(k+2)!} = \frac{e^z - 1 - z}{z^2} \tag{A.3}$$

Equation A.3 has the general form as follows:

$$E_{1,m}(z) = \frac{1}{z^m - 1}(e^z - \sum_{k=0}^{m-2} \frac{z^k}{k!}) \tag{A.4}$$

Mittag-Leffler function's relation to other functions can be given with the following equations [12]:

$$E_{1,1}(z) = e^z, \tag{A.5}$$

$$E_{1,2}(z) = \frac{e^z - 1}{z}, \tag{A.6}$$

$$E_{2,1}(z) = \cosh(\sqrt{z}), \tag{A.7}$$

$$E_{2,2}(z) = \frac{\sinh(\sqrt{z})}{\sqrt{z}}, \tag{A.8}$$

$$E_{1/2,1}(\sqrt{z}) = \frac{2}{\sqrt{\Pi}} e^{-z} erfc(-\sqrt{z}) \tag{A.9}$$

where erfc in A.9 is the *error function complement* defined by [10]:

$$erfc(z) = \frac{2}{\sqrt{\Pi}} \int_z^{\infty} e^{-t^2} dt. \tag{A.10}$$

A.2. Beta Function

The Beta Function is defined by:

$$B(z,w) = \int_0^1 \tau^{z-1}(1-\tau)^{w-1} d\tau, hspace.1cm(\Re e(z) > 0, hspace.1cm \Re e(w) > 0) \quad \text{(A.11)}$$

With the help of Beta Function, the following two important relations can be established for the Gamma Function. The first one is as follows:

$$\Gamma(z)\Gamma(1-z) = \frac{\Pi}{\sin(\Pi z)} \quad \text{(A.12)}$$

The second one is the Legendre formula:

$$\Gamma(z)\Gamma(z+\frac{1}{2}) = \sqrt{\Pi}2^{2z-1}\Gamma(2z), \ \ (2z \neq 0, -1, -2, \ldots) \quad \text{(A.13)}$$

A.3. Cauchy's Integral Formula

$$f(z_0) = \frac{1}{2\Pi i} \oint_\gamma \frac{f(z)dz}{z - z_0} \quad \text{(A.14)}$$

REFERENCES

1. Kilbas, A. A., H. M. Srivastava, and J. J. Trujillo, "Theory and Applications of Fractional Differential Equations", Elsevier, Amsterdam, 2006.

2. Podlubny, I., "Fractional-Order Systems and $PI^{\lambda}D^{\mu}$ Controllers", *IEEE Transections on Automatic Control*, Vol. 44, No. 1, pp. 208-214, 1999.

3. Petráš, I., "The Fractional-Order Controllers: Methods for Their Synthesis and Application", *J. Electrical Eng.*, Vol. 50, No. 9-10, pp. 284-288, 1999.

4. Zeng, Q. -S., C. Guang-Yi, and X. -J. ZhuThe, "Effect of The Fractional-Order Controller's Orders Variation on The Fractional-Order Control Systems", *in Proc. of IEEE 2003 the First International Conference on Machine Learning and Cybernetics*, pp. 367-372, Beijing, November 2002.

5. Kennedy, J. and R. Eberhart, "Particle Swarm Optimization", *in Proc. of the IEEE International Conference on Neural Networks*, pp. 1942-1948, Piscataway, NJ, 1995.

6. Maiti, D. and A. Konar, "Approximation of a Fractional Order System by an Integer Order Model Using Particle Swarm Optimization Technique", *in Proc. of the IEEE Conference on Computational Intelligence, Control and Computer Vision in Robotics and Automation, (CICCRA 08)*, pp. 149-152, NIT Rourkela, 2008.

7. Maiti D., S. Biswas, and A. Konar, "Design of Fractional Order PID Controller Using Particle Swarm Optimization Technique", *in Proc. of 2nd National Conference on Recent Trends in Information Systems, (RETIS 08)*, Kolakata, 2008.

8. Das, S., "Functional Fractional Calculus for System Identification and Controls", Springer-Verlag, Germany, 2008.

9. Ross, B., "Fractional Calculus: An Historical Apologia for The Development of a

Calculus Using Differentiation and Antidifferentiation of Non-integral Orders", *Mathematics Magazine*, Vol. 50, No. 3, pp. 115-122, May 1977.

10. Podlubny, I., "Fractional Differential Equations: An Introduction to Fractional Derivatives, Fractional Differential Equations, to Methods of their Solution and some of their Applications", *Mathematics in Science and Engineering*, Vol. 198, Academic Press, USA, 1999.

11. Davis, H. D., "The Theory of Linear Operators", Principia Press, USA, 1936.

12. Podlubny, I., "Laplace Transform Method for Linear Differential Equations of The Fractional Order", *Slovak Academy of Sciences, Institute of Experimental Physics*, Slovak Republic, 1994.

13. Hashim, I. and O. Abdulaziz, "Homotopy Analysis Method for Fractional IVPs", *Communications in Nonlinear Science and Numerical Simulation* Vol. 14, No. 3, pp. 674-684, March 2009.

14. Kilbas, A.A., M. Rivero, L. Rodríguez-Germá, and J. J. Trujillo, "α-Analytic Solutions of Some Linear Fractional Differential Equations with Variable Coefficients, *Applied Mathematics and Computation*, Vol. 187, No. 1, pp. 239-249, April 2007.

15. Bonilla, B., M. Rivero, and J.J. Trujillo, "On Systems of Linear Fractional Differential Equations with Constant Coefficients" , *Applied Mathematics and Computation*, Vol. 187, No. 1, pp. 68-78, April 2007.

16. Odibat, Z., S. Momani, and V. S. Erturk, "Generalized Differential Transform Method: Application to Differential Equations of Fractional Order", *Applied Mathematics and Computation*, Vol. 197, No. 2, pp. 467-477, April 2008.

17. Shokooh, A. and L. Suarez, "A Comparison of Numerical Methods Applied to A Fractional Model of Damping Materials", *Journal of Vibration and Control*, Vol. 5, No. 3, pp. 331-354, 1999.

18. Podlubny, I., "Matrix Approach to Discrete Fractional Calculus", *Fractional Calculus and Applied Analysis*, Vol. 3, No. 4, pp. 359-386, 2000.

19. Gejji, V.D. and H. Jafari, "Solving A Multi-Order Fractional Differential Equation Using Adomian Decomposition", *Applied Mathematics and Computation*, Vol. 189, No. 1, pp. 541-548, June 2007.

20. Dorčák, L., " Numerical Models for The Simulation of The Fractional-Order Control Systems", *Slovak Academy of Sciences Institute of Experimental Physics*, Slovakia, November 1994.

21. Podlubny, I., "Geometric and Physical Interpretation of Fractional Integration and Fractional Differentiation, *Fractional Calculus and Applied Analysis: An International Journal for Theory and Applications*, Vol. 5, No.4, pp. 367-386, 2002.

22. Merrikh-Bayat, F. and M. Karimi-Ghartemani, "On The Essential Instabilities Caused by Fractional-Order Transfer Functions", *Mathematical Problems in Engineering*, Vol. 2008, pp. 1-13, 2008.

23. Hartley, T. T. and C. F. Lorenzo, "A Solution to The Fundamental Linear Fractional-Order Differential Equation", *National Aeronautics and Space Administration (NASA)-Lewis Research Center*, December 1998.

24. Silva, G. J., A. Datta, and S. P. Bhattacharya, "PID Controllers for Time-Delay Systems", Birkhäuser, Boston, 2005.

25. Ogata, K., "Modern Control Engineering Third Edition", Prentice Hall, New Jersey, 1997.

26. Viangre, B. M., I. Podlubny, L. Dorčák, V. Feliu, "On Fractional-PID Controllers:A Frequency Domain Approach, *Proc. of IFAC Workshop on Digital Control. Past, Present and Future of PID Control*, pp. 53-58, Terrasa, 2000.

27. Merrikh-Bayat, F. and M. Afshar, "Extending The Root-Locus Method to

Fractional-Order Systems", *Journal of Applied Mathematics*, Vol. 2008, pp. 1-13, 2008.

28. Barbosa, R. S., J. A. T. Machado, and I. M. Ferreira., "A Fractional Calculus Perspective of PID Tuning", *Proc. of ASME 2003 Design Engineering Technical Conferences (DETC 2003) and Computers and Information in Engineering Conference*, pp. 1-9, Chicago, July 2005.

29. Zhao, C., D. Xue, and Y. Chen., "A Fractional Order PID Tuning Algorithm for a Class of Fractional Order Plants", *Proc. of the IEEE International Conference on Mechatronics and Automation*, pp. 216-221, Niagara Falls, September 2003.

30. Vinagre, B. M., C. A. Monje, and A. J. Calderón, J. I. Suárez, "Fractional PID Controllers for Industry Application: A Brief Introduction", *Journal of Vibration and Control*, Vol. 13, No. 9-10, pp. 1419-1429, July 2007.

31. Monje, C. A., B. M. Vinagre, Y. Chen, and V. Feliu, "On Fractional PI^{λ} Controllers: Some Tuning Rules for Robustness to Plant Uncertainties", *Nonlinear Dynamics*, Springer Netherlands, Vol. 38, No. 1-2, pp. 369-381, December 2004.

32. Maiti, D., A. Acharya, M. Chakraborty, A. Konar, and R. Janarthanan, "Tuning PID and $PI^{\lambda}D^{\delta}$ Controllers Using The Integral Time Absolute Error Criterion", *Proc. of International Conference on Information and Automation for Sustainability-ICIAfS*, 2008.

33. Podlubny, I. "Fractional-Order Systems and Fractional-Order Controllers", *The Academi of Sciences Institute of Experimental Physics*, UEF-03-94, Kosice, Slovak Republic; 1994.

34. Miller, K. S. and Ross, B.,"An Introduction to The Fractional Calculus and Fractional Differntial Equations", Wiley-Interscience, USA, 1993.