Design of Fractional Order $PI^{\lambda}D^{\mu}$ Controllers with an Improved Differential Evolution

Ajith Abraham
Q2S, NTNU
Trondheim, Norway
ajith.abraham@ieee.org

Arijit Biswas
Jadavpur University
Kolkata, India
arijitbiswas87@gmail.com

Swagatam Das
Jadavpur University,
Kolkata, India
sudurero.piyasi@gmail.com

Sambarta Dasgupta
Jadavpur University
Kolkata, India
sambartadg@gmail.com

ABSTRACT
Differential Evolution (DE) has recently emerged as a simple yet very powerful technique for real parameter optimization. This article describes an application of DE for the design of Fractional-Order Proportional-Integral-Derivative (FOPID) Controllers involving fractional order integrator and fractional order differentiator. FOPID controllers' parameters are composed of the proportionality constant, integral constant, derivative constant, derivative order and integral order, and its design is more complex than that of conventional integer order PID controller. Here the controller synthesis is based on user-specified peak overshoot and rise time and has been formulated as a single objective optimization problem. In order to digitally realize the fractional order closed loop transfer function of the designed plant, Tustin operator-based CFE (continued fraction expansion) scheme was used in this work. Simulation examples as well as comparisons of DE with two other state-of-the-art optimization techniques (Particle Swarm Optimization and Bacterial Foraging Optimization Algorithm) over the same problems demonstrate the superiority of the proposed approach especially for actuating fractional order plants.

Categories and Subject Descriptors
I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search -- *Heuristic methods*; G.1.6 [**Numerical Analysis**]: Optimization -- *Global optimization*

General Terms
Design

Keywords
Fractional Order Controller, PID controller, Differential Evolution, Swarm Intelligence.

1. INTRODUCTION
Fractional order dynamic systems and controllers, which are based on fractional order calculus [1-3], have been gaining attention in several research communities since the last few years [4, 5].

GECCO'08, July 12--16, 2008, Atlanta, Georgia, USA.

Podlubny [6] proposed the concept of the fractional order $PI^{\lambda}D^{\mu}$ Controllers and demonstrated the effectiveness of such controllers for actuating the responses of fractional order systems in 1999. A few recent works in this direction as well as schemes for digital and hardware realizations of such systems can be traced in [7-9]. Vinagre *et al.* [10] proposed a frequency domain approach based on expected crossover frequency and phase margin for the same controller design. Petras [11] proposed w a method based on the pole distribution of the characteristic equation in the complex plane. Dorcak *et al.* [12] proposed a state space design approach based on feedback pole placement. The fractional controller can also be synthesized by cascading a proper fractional unit to an integer order controller [5].

PID controllers have been used for several decades in industries for process control applications .The reason for their wide popularity lies in the simplicity of design and good performance including low percentage overshoot and small settling time for slow process plants [13]. In FOPID controller *I* and *D* operations are usually of fractional order, therefore besides setting the proportional, derivative and integral constants K_p, T_d, T_i we have two more parameters: the order of fractional integration λ and that of fractional derivative μ . Finding an optimal set of values for $K_p, T_i, T_d, \lambda,$ and μ to meet the user specifications for a given process-plant calls for real parameter optimization in five-dimensional hyperspace.

Differential Evolution (DE) [14, 15] has recently become quite popular as a simple and efficient scheme for global optimization over continuous spaces. It has reportedly outperformed many types of evolutionary algorithms and search heuristics like Particle Swarm Optimization (PSO) when tested over both benchmarks and real world problems [16]. In this research, a state-of-the-art version of DE has been used for finding the optimal values of $K_p, T_i, T_d, \lambda,$ and μ .The design method focuses on optimum placing of the dominant closed loop poles and incorporate the constraints thus obtained using DE algorithm. The optimization-based design process has been tested for actuating the response of two process plants of which one is of integer order and the other is of fractional order. The performance of the DE based $PI^{\lambda}D^{\mu}$ controller has been compared with two other fractional order controllers designed with the state of the art versions of two recent population based techniques well known as the HPSO-TVAC [17] and the

Genetic Algorithm [18,19]. Such comparison reflects the superiority of the proposed method in terms of quality of the final solution, convergence speed and robustness.

The rest of the paper is organized in the following way. Section 2 describes the rudiments of fractional calculus and fractional order control systems. Section 3 provides a brief overview of the DE family of algorithms and describes a recent state of the art version of DE called DE/rand/either-or, which was used, in this specific task. Section 4 demonstrates how the DE can be applied to the $PI^{\lambda}D^{\mu}$ controller design problem when formulated as an optimization task. Simulation strategies and experimental results has been presented and discussed in Section 5 and finally the paper is concluded with a discussion on future research issues in Section 6.

2. FRACTIONAL ORDER SYSTEMS: A BRIEF OVERVIEW

Fractional calculus is a branch of mathematical analysis that studies the possibility of taking real number power of the differential operator and integration operator. From a purely mathematical point of view there are several ways to define fractional order derivatives and integrals. The generalized differintegrator operator is given as:

$$_aD_t^q f(t) = \frac{d^q f(t)}{[d(t-a)]^q} \quad (1)$$

Where q represents the real order of the differintegral (an n is used in some literature to denote an integer order), t is the parameter for which the differintegral is taken, and a is the lower limit. Unless otherwise stated the lower limit will be *0* and left out of the notation. Caputo used a popular definition used to compute differintegral in 1960s. The definition for Caputo's fractional derivative of order λ with respect to the variable t and with the starting point $t = 0$ goes as follows [20,21]:

$$_0D_t^{\lambda} y(t) = \frac{1}{\Gamma(1-\delta)} \int_0^t \frac{y^{(m+1)}(\tau)d\tau}{(t-\tau)^{\delta}}, \quad (2)$$
$$(\gamma = m+\delta, m \in Z, 0 < \delta \le 1)$$

Where $\Gamma(z)$ is Euler's Gamma function. If $\gamma < 0$, then we have a fractional integral of order $-\gamma$ given as:

$$_0I_t^{-\gamma} y(t) = {}_0D_t^{\gamma} y(t) = \frac{1}{\Gamma(-\gamma)} \int_0^t \frac{y(\tau)d\tau}{(t-\tau)^{1+\gamma}}, (\gamma < 0) \quad (3)$$

One distinct advantage of using the Caputo's definition is that it only allows for consideration of easily interpretable initial conditions but it is also bounded, which means the derivative of a constant is equal to zero. In time domain, a fractional order system is governed by an n-term inhomogeneous fractional order differential equation (FDE):

$$a_n D^{\beta_n} y(t) + a_{n-1} D^{\beta_{n-1}} y(t) + \\ + a_1 D^{\beta_1} y(t) + a_0 D^{\beta_0} y(t) = u(t) \quad (4)$$

Where $D^{\lambda} \equiv {}_0D_t^{\lambda}$ is the Caputo's fractional derivative of order λ. Converting to frequency domain, the fractional order transfer function of such a system may be obtained through the Laplace transform function as follows,

$$G_n(s) = \frac{1}{a_n s^{\beta_n} + a_{n-1} s^{\beta_{n-1}} + ...+ a_1 s^{\beta_1} + a_0 s^{\beta_0}} \quad (5)$$

Where $\beta_k (k = 0,1,......., n)$ is an arbitrary real number, $\beta_n > \beta_{n-1} > .... > \beta_1 > \beta_0 > 0$

and $a_k (k = 0,1,......., n)$ is an arbitrary constant. Finally we would like to mention here that the Laplace transform of the fractional derivative might be given as,

$$\int_0^{\infty} e^{-st} D^{\gamma} y(t)dt = s^{\gamma} Y(s) - \sum_{k=0}^{m} s^{\gamma-k-1} y^{(k)}(\theta) \quad (6)$$

For $\gamma < 0$, (i.e., for the case of a fractional integral) the sum in the right hand side must be omitted.

3. THE DE ALGORITHM AND ITS MODIFICATION

Like any other evolutionary algorithm, DE starts with a population of *NP D*-dimensional parameter vectors representing the candidate solutions. We shall denote subsequent generations in DE by $G = 0,1..., G_{max}$. Since the parameter vectors are likely to be changed over different generations we may adopt the following notation for representing the *i*-th vector of the population at the current generation as:

$$\vec{X}_{i,G} = [x_{1,i,G}, x_{2,i,G}, x_{3,i,G}, ..., x_{D,i,G}]. \quad (7)$$

The initial population (at $G = 0$) should better cover the entire search space as much as possible by uniformly randomizing individuals within the search space constrained by the prescribed minimum and maximum bounds: $\vec{X}_{min} = \{x_{1,min}, x_{2,min}, ..., x_{D,min}\}$ and $\vec{X}_{max} = \{x_{1,max}, x_{2,max}, ..., x_{D,max}\}$. Hence we may initialize the *j*-th component of the *i*-th vector as:

$$x_{j,i,0} = x_{j,min} + rand_j(0,1).(x_{j,max} - x_{j,min}) \quad (8)$$

where $rand_j(0,1)$ is the *j*-th instantiation of a uniformly distributed random number lying between 0 and 1. Following steps are taken next: mutation, crossover, and selection, which are explained below.

a) Mutation

After initialization, DE creates a *donor* vector $\vec{V}_{i,G}$ corresponding to each population member or *target* vector $\vec{X}_{i,G}$ in the current generation through mutation. It is the method of creating this donor vector, which differentiates between the various DE schemes. For example, five most frequently referred DE mutation strategies implemented and available in the public-domain [26] are listed below:

DE/rand/1: $\vec{V}_{i,G} = \vec{X}_{r_1^i,G} + F.(\vec{X}_{r_2^i,G} - \vec{X}_{r_3^i,G})$ (9)

DE/best/1: $\vec{V}_{i,G} = \vec{X}_{best,G} + F.(\vec{X}_{r_1^i,G} - \vec{X}_{r_2^i,G})$ (10)

DE/target-to-best/1:

$$\vec{V}_{i,G} = \vec{X}_{i,G} + F.(\vec{X}_{best,G} - \vec{X}_{i,G}) + F.(\vec{X}_{r_1^i,G} - \vec{X}_{r_2^i,G}) \quad (11)$$

DE/best/2: $\vec{V}_{i,G} = \vec{X}_{best,G} + F.(\vec{X}_{r_1^i,G} - \vec{X}_{r_2^i,G}) + F.(\vec{X}_{r_3^i,G} - \vec{X}_{r_4^i,G})$ (12)

DE/rand/2: $\vec{V}_{i,G} = \vec{X}_{r_1^i,G} + F.(\vec{X}_{r_2^i,G} - \vec{X}_{r_3^i,G}) + F.(\vec{X}_{r_4^i,G} - \vec{X}_{r_5^i,G})$ (13)

The indices r_1^i, r_2^i, r_3^i, r_4^i, and r_5^i are mutually exclusive integers randomly chosen from the range [1, *NP*], which are also different from the index *i*. These indices are randomly generated once for each mutant vector. The scaling factor *F* is a positive control parameter for scaling the difference vectors. $\vec{X}_{best,G}$ is the best individual vector with the best fitness function value in the population at generation *G*. The general convention used for naming the various mutation strategies is DE/x/y/z, where DE stands for Differential Evolution, x represents a string denoting the vector to be perturbed and y is the number of difference vectors considered for perturbation of x. z stands for the type of crossover being used (exp: exponential; bin: binomial).

b) Crossover:

To increase the potential diversity of the population, a crossover operation comes into play after generating the donor vector through mutation. The classical DE family of algorithms generally uses two kinds of crossover schemes - *exponential* and *binomial* [14]. The donor vector exchanges its components with the target vector $\vec{X}_{i,G}$ under this operation to form the *trial* vector $\vec{U}_{i,G} = [u_{1,i,G}, u_{2,i,G}, u_{3,i,G}, ..., u_{D,i,G}]$. Here we briefly discuss the binomial crossover and the arithmetic crossover, which has recently been introduced in the DE community in order to circumvent the problem of rotational variance. The binomial crossover is performed on each of the *D* variables whenever a randomly picked number between 0 and 1 is less than or equal to the *Cr* value. In this case the number of parameters inherited from the mutant has a (nearly) binomial distribution. The scheme may be outlined as,

$$u_{j,i,G} = v_{j,i,G}, \text{ if } (rand_j(0,1) \le Cr \text{ or } j = j_{rand}$$

$$= x_{j,i,G}, \text{ otherwise} \quad (14)$$

where $rand_j(0,1) \in [0,1]$ is the *j*-th evaluation of a uniform random number generator. $j_{rand} \in [1,2,...., D]$ is a randomly chosen index, which ensures that $\vec{U}_{i,G}$ gets at least one component from $\vec{V}_{i,G}$.

The crossover scheme described in (14) is in spirit a discrete recombination [14]. The discrete recombination is a rotationally variant operation. A rotation of the coordinate systems moves the location of the potential trial solutions. To overcome this limitation, a new trial vector generation strategy 'DE/current-to-rand/1' is proposed in [22], which replaces the crossover operator prescribed in equation (14) with the rotationally invariant arithmetic crossover operator to generate the trial vector $\vec{U}_{i,G}$ by linearly combining the target vector $\vec{X}_{i,G}$ and the corresponding donor vector $\vec{V}_{i,G}$ as follows:

$$\vec{U}_{i,G} = \vec{X}_{i,G} + K.(\vec{V}_{i,G} - \vec{X}_{i,G}). \quad (15)$$

Now incorporating (9) in (15) we have:

$$\vec{U}_{i,G} = \vec{X}_{i,G} + K.(\vec{X}_{r_1,G} + F.(\vec{X}_{r_2,G} - \vec{X}_{r_3,G}) - \vec{X}_{i,G}),$$

which further simplifies to:

$$\vec{U}_{i,G} = \vec{X}_{i,G} + K.(\vec{X}_{r_1,G} - \vec{X}_{i,G}) + F'.(\vec{X}_{r_2,G} - \vec{X}_{r_3,G}) \quad (16)$$

where *K* is the combination coefficient, which has been proven [22] to be effective when it is chosen with a uniform random distribution from [0, 1] and *F' = K.F* is a new constant here.

c) Selection:

To keep the population size constant over subsequent generations, the next step of the algorithm calls for *selection* to determine whether the target or the trial vector survives to the next generation i.e. at $G = G + 1$. The selection operation may be outlined as:

$$\vec{X}_{i,G+1} = \vec{U}_{i,G}, \quad \text{if } f(\vec{U}_{i,G}) \le f(\vec{X}_{i,G})$$

$$= \vec{X}_{i,G}, \quad \text{if } f(\vec{U}_{i,G}) > f(\vec{X}_{i,G}) \quad (17)$$

where $f(\vec{X})$ is the function to be minimized. So if the new trial vector yields a lower value of the objective function, it replaces the corresponding target vector in the next generation; otherwise the target is retained in the population. Hence the population either gets better (with respect to the minimization of the objective function) or remains constant, but never deteriorates.

In the original DE mutation scheme, the difference vector ($\vec{X}_i(t)$ - $\vec{X}_j(t)$) is scaled by a constant factor '*F*'. The usual choice for this control parameter is a number between 0.4 and 1. We propose to vary this scale factor in a random manner in the range (0.5, 1) by using the relation

$$F = 0.5 * (1 + rand(0,1)) \quad (18)$$

where *rand (0, 1)* is a uniformly distributed random number within the range [0, 1]. The mean value of the scale factor is 0.75. This allows for stochastic variations in the amplification of the difference vector and thus helps retain population diversity as the search progresses. Das *et al.* [23] has illustrated the DERANDSF (DE with Random Scale Factor) can outperform the classical DE and also some versions of PSO in a statistically significant manner. In addition to that, here we also decrease the crossover rate CR linearly with time from CR_{max} = 1.0 to CR_{min} = 0.5. If *CR* = 1.0, it means that all components of the parent vector are replaced by the difference vector operator according to (14). But at the later stages of the optimizing process, if *CR* be decreased, more components of the parent vector are then inherited by the offspring. Such a tuning of *CR* helps to explore the search space exhaustively at the beginning, but adjust the movements of trial solutions finely during the later stages of search, so that they can explore the interior of a relatively small space in which the suspected global optimum lies. The time-variation of CR may be expressed in the form of the following equation:

$$CR = (CR_{max} - CR_{min}) * (\frac{G_{max} - G}{G_{max}}) + CR_{min} \quad (19)$$

where CR_{max} and CR_{min} are the maximum and minimum values of crossover rate CR, G is the current generation number and G_{max} is the maximum number of allowable generations. After performing a series of experiments we find that the DE/rand/1/bin scheme (9) equipped with these modifications can outperform all other classical DE variants for the controller design problem investigated here.

4. THE DE-BASED DESIGN OF FRACTIONAL PI$^{\lambda}$D$^{\mu}$ CONTROLLERS

4.1 The FOPID Controller

A PID controller is a generic control loop feedback mechanism widely used in industrial control systems. The PID controller attempts to correct the error between a measured process variable and a desired set point by calculating and then outputting a corrective action that can adjust the process accordingly. An integer order PID controller has the following transfer function:

$$G_c(s) = K_p + K_i s^{-1} + K_d s \quad (20)$$

The PID controller calculation (algorithm) involves three separate parameters; the Proportional (K_p), the Integral (K_i) and Derivative (K_d) time-constants. The Proportional gain determines the reaction to the current error, the Integral determines the reaction based on the sum of recent errors and the derivative determines the reaction to the rate at which the error has been changing. The weighted sum of these three actions is used to adjust the process via a control element such as the position of a control valve or the power supply of a heating element. The block diagram of a generic closed loop control system with the PID controller is illustrated in Figure 1.

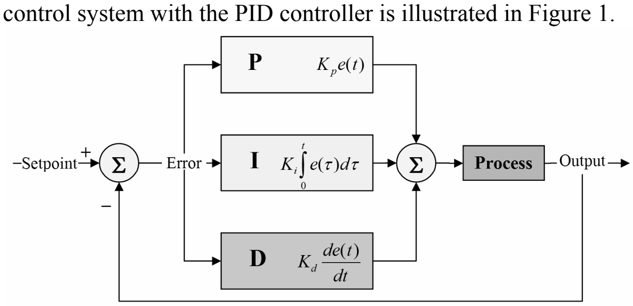


Figure 1. A generic closed-loop process-control system with PID controller

The real objects or processes that we want to control are generally fractional (for example, the voltage-current relation of a semi-infinite lossy RC line). However, for many of them the fractionality is very low. In general, the integer-order approximation of the fractional systems can cause significant differences between mathematical model and real system. The main reason for using integer-order models was the absence of solution methods for fractional-order differential equations. PID controllers belong to dominating industrial controllers and therefore are objects of steady effort for improvements of their quality and robustness. One of the possibilities to improve PID controllers is to use fractional-order controllers with non-integer derivation and integration parts.

Following the works of Podlubny [6] we may go for a generalization of the PID-controller, which can be called the PI$^{\lambda}$D$^{\mu}$ -controller because of involving an integrator of order λ and a differentiator of order μ. The continuous transfer function of such a controller has the form:

$$G_c(s) = K_p + T_i s^{-\lambda} + T_d s^{\mu}, (\lambda, \mu > 0) \quad (21)$$

The output response of the PI$^{\lambda}$D$^{\mu}$ -controller in time domain may be given as:

$$u(t) = K_p . e(t) + K_i . D^{-\lambda} e(t) + K_d . D^{\mu} e(t) \quad (22)$$

Where, $\lambda = +1, \mu = +1$ implies normal PID controller, for $\lambda = 0, \mu = +1$, we get a normal PD controller, $\lambda = +1, \mu = 0$ implies normal PI controller and $\lambda = 0, \mu = 0$ implies a proportional gain. All these classical types of PID-controllers are the special cases of the fractional PI$^{\lambda}$D$^{\mu}$ -controller. As depicted in Figure 2, the fractional order PID controller generalizes the integer order PID controller and expands it from point to plane. This expansion adds more flexibility to controller design and we can control our real world processes more accurately.

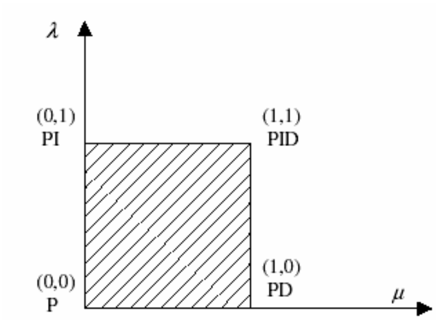


Figure 2. Generalization of the FOPID Controller: From point to plane

4.2 Formulation of the Objective Function

The design approach presented here is based on the root locus method (dominant roots method) of synthesizing integral PID controllers [13]. As in the traditional root locus method, based on the user specifications of peak overshoot M_p and rise time t_{rise} (or requirements of stability and damping levels), we find out the damping ratio ξ and the un-damped natural frequency ω_o of the closed loop system to be designed. Then dominant poles will be:

$$p_{1,2} = -\xi\omega_0 \pm j\omega_0\sqrt{1-\xi^2} = -x \pm jy \quad (23)$$

Let the closed loop transfer function be:

$$\frac{C(s)}{R(s)} = \frac{G(s)}{1+G(s)H(s)} \quad (24)$$

where the transfer function of the process to be controlled is $G_p(s)$ and that of the controller is $G_c(s) = \frac{U(s)}{E(s)}$ and $G(s) = G_c(s)G_P(s)$. We assume unity feedback gain i.e. $H(s) = 1$. From (24) the characteristics equation of the closed loop system is given by:

$$1 + G(s)H(s) = 0, \quad \Rightarrow 1 + G_p(s)G_c(s).1 = 0 \quad (25)$$

Now the dominant poles of the system are the zeros of this characteristics equation, so they will obviously satisfy that equation. Thus from (25) we get,

$$1+[K_p+K_i(-x+jy)^{-\lambda}+K_d(-x+jy)^{\mu}].G_p(-x+jy)=0. \quad (26)$$

This equation has total five unknowns, $K_p, K_i, K_d, \lambda, \mu$. Let:

R =Real part of the complex expression (26),
I =Imaginary part of the complex expression (26), and
ψ =Phase angle = $\tan^{-1}(I/R)$.

Now we define the following objective function:

$$J(K_p, K_i, K_d, \lambda, \mu) = |I|^2 + |R|^2 + |\psi|^2 \quad (27)$$

Our goal is to find out an optimal solution set $\{K_p, K_i, K_d, \lambda, \mu\}$ for which $J = 0$. Here the above function has been minimized with DE/rand/1/either-or algorithm.

4.3 Vector Representation in DE

The solution space of equation (27) is 5-dimensional, the five dimensions being $\{K_p, K_i, K_d, \lambda, \mu\}$. So each parameter vector in DE has 5 components i.e. the *j*-th population member at *G*-th generation may be given as:

$$\vec{X}_{j,G} = (K_p, K_i, K_d, \lambda, \mu)^T \quad (28)$$

From the practical consideration of the PID controller design [13], we fixed the following numerical ranges for each parameter:

$$1 \le K_p \le 1000$$
$$0 \le \lambda, \delta \le 1 \quad (29)$$
$$1 \le T_i, T_d \le 500$$

Table 1. Description of the problem instances considered

Problem Number	Process Plant Transfer Function $G_p(s)$	Users specification		
		Maximum overshoot (%)	Rise time (Sec)	Steady state error (%)
I	$\frac{k}{(Js+b)(Ls+R)+k^2}$ $J=0.01, b=0.1,$ $k=0.01,$ $R=1, L=0.5,$	5	0.5	4
II	$\frac{1}{0.9s^{0.3}+0.6s^{0.8}+1}$ (fractional plant)	5	0.3	3

5. EXPERIMENTAL RESULTS

5.1 Problem Instances

We have tested the proposed method on two specific instances of the design problem. All the design examples follow the basic framework detailed in Section 4. The first problem involves the speed control of a DC motor. First, the uncompensated motor can only rotate at 0.1 rad/sec with an input voltage of 1 Volt (this was obtained when the open-loop response is simulated). Since the most basic requirement of a motor is that it should rotate at the desired speed, the steady-state error of the motor speed should be less than 1%. The other performance requirement is that the motor must accelerate to its steady-state speed as soon as it turns on. In this case, we want it to have a settling time of 2 seconds. Since a speed faster than the reference may damage the equipment, we want to have an overshoot of less than 5%.
The second problem instance involve a fractional order plant. In some cases a real system is better described by such fractional order differential equations [24] and from this consideration, it is important to investigate the controlling mechanism of such systems through FOPID type controllers. Table 1 summarizes the test problems along with the corresponding user specifications.

5.2 Digital Realization of the FOPID Controller

For a fractional order differentiator/integrator s^r , where r is a real number, its discretization is a key step in digital implementation. Furthermore for control applications, obtained approximate discrete time rational transfer function should be stable and of minimum phase. Continuous fraction expansion (CFE) by Tustin rule enjoys all those desirable properties. By using this method the discrete transfer function approximating fractional order operators can be expressed as:

$$D^{\pm r}(z) = (w(z^{-1}))^{\pm r} = \left(\frac{2}{T}\right)^{\pm r} CFE\left(\left(\frac{1-z^{-1}}{1+z^{-1}}\right)^{\pm r}\right)_{p,q} = \left(\frac{2}{T}\right)^{\pm r} \frac{P_p(z^{-1})}{Q_q(z^{-1})} \quad (30)$$

Where T is the sampling period and P_p and Q_q are polynomials of degree p and q , respectively, in the variable z^{-1} . The general expression for numerator $P_p(z^{-1})$ and denominator $Q_q(z^{-1})$ of $D^{\pm r}(z)$ is given below for $p = q = 1,3,5$. Expressions for numerator and denominator polynomials in the CFE are summarized in Table 2.

In this work we have used the Tustin rule based CFE where the sampling time is $T = 0.001s$ and the order of the approximate model is 5.

Table 2. Expressions for numerator and denominator polynomials in the CFE

p=q	$P_p(z^{-1})$ $(k=1)$ and $Q_q(z^{-1})$ $(k=0)$
1	$(-1)^k z^{-1} r + 1$
3	$(-1)^k (r^3 - 4r) z^{-3} +$ $(6r^2 - 9) z^{-2} +$ $(-1)^k 15 z^{-1} r + 15$
5	$(-1)^k (r^5 - 20r^3 + 64r) z^{-5} +$ $(-195r^2 + 15r^4 + 225) z^{-4} +$ $(-1)^k (105r^3 - 735r) z^{-3} +$ $(420r^2 - 1050) z^{-2} + (-1)^k 945 z^{-1} r + 945$

Table 3. Parameter settings for the different algorithms

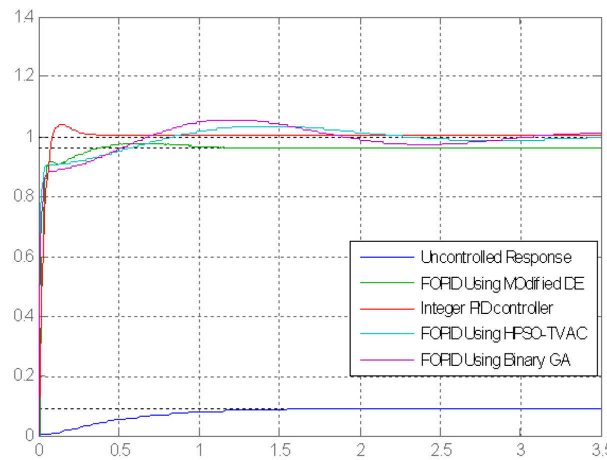
HPSO-TVAC		Modified DE		Binary GA [16]	
Parameter	Value	Parameter	Value	Parameter	Value
Pop size	40	Pop size	10**D*	Initial Pop size	50
Inertia weight	0.794	CR_{max}	1.0	No. of bits per gene	50
C_1	Linearly varying 0.35→2.4	CR_{min}	0.5	Mutation probability	0.01
C_2	Linearly varying 2.4→0.35	Scale factor F	Uniformly distributed random number between 0.5 and 1.0 with mean value 0.75	Uniform crossover probability	0.6
V_{max}	3.00				
Re-initialization velocity	Linearly decaying from V_{max} to 0.1 V_{max}				

5.3 Algorithms and Parametric Set-up

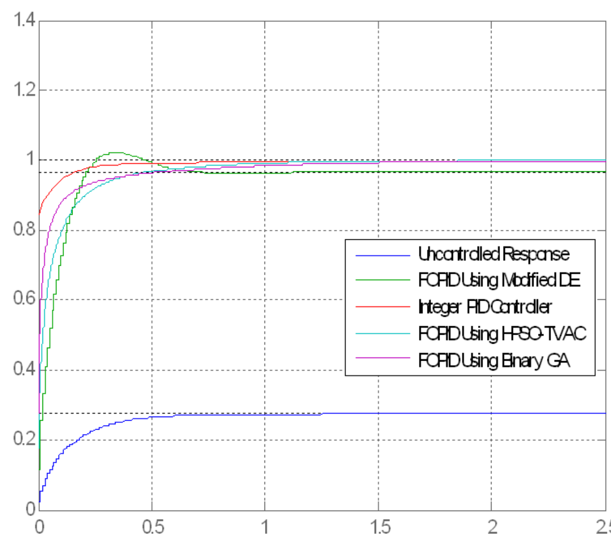
The proposed design method has been extensively compared with two state-of-the-art design methods for FOPID controllers based on the binary GA [18,19] and a recently proposed extension of the canonical PSO namely Self Organizing Hierarchical Particle Swarm Optimizer with Time Varying Acceleration Coefficients (HPSO-TVAC) [17]. The GA based scheme was proposed by Cao *et al.* [19] and uses a 50 bits binary string to encode 5 parameters of the FOPID controller. The fitness-functions in Cao's method employ the integral of the squared error and absolute error signal value and are typically borrowed from the realm of optimal control [25]. The HPSO-TVAC algorithm on the other hand uses the same particle representation scheme as well as objective function as that used for the modified DE. Table 3 illustrates the parametric set-up for these algorithms. We choose the standard set of parameters, equipped with which, the algorithms have been shown to be at the peak of their performance (over benchmark functions) in the existing literature [17, 19, and 23]. No hand tuning of parameters have been allowed in any case to make the comparison fair enough.

5.4 Simulation Strategy

We run three population based optimization algorithms namely HPSO-TVAC, a modified DE, and the binary encoded GA suggested in [18,19] for the two design problems according to the user specifications summarized in Table 1. All the algorithms have been developed in Visual C++ platform on a Pentium IV, 2.2 GHz PC, with 512 KB cache and 2 GB of main memory in Windows Server 2003 environment. 25 independent runs (with different seeds for the random number generator) were carried out for each of the algorithms and each run was continued up to 10^5 Function Evaluations (FEs). In case of DE since D = 5, NP = 50 and this approximately corresponds to a $G_{\max} = 2000$ for 10^5 FEs. We report the empirical results for the median run of each algorithm (when the runs for a single algorithm have been ranked according to their final accuracy).



(a) Design Problem 1 (integer order plant)



(b) Design Problem 4 (Fractional Order Plant)

Figure 3. Unit step response of the closed loop systems for the test problems

Table 5. Summary of the performance of closed loop system under different PID controllers against the unit step response

Process Plant	Different controllers used	Unit step response obtained: Maximum overshoot (%) ± standard deviation (%)	Rise time (Sec) ± standard deviation (Sec)	Steady state error (%) ± standard deviation (%)	Final objective function values obtained
I	Fractional Controller using DE	3.11 ± (0.31)	0.395 ±(0.051)	3.5 ±(0.010)	0.00 ±(0.0000)
	Integer PID controller using DE	4.23 ± (0.34)	0.101 ±(0.007)	0.1 ±(0.001)	0.00 ±(0.0000)
	Fractional Controller using PSO	3.91 ± (0.41)	0.822 ±(0.091)	1. 9 ±(0.021)	0.0001 ±(0.0000)
	Fractional Controller using GA	6.31 ± (0.87)	0.695 ±(0.088)	2.1 ±(0.056)	0.0312 ±(0.0025)
II	Fractional Controller using DE	1.93 ±(0.089)	0.218 ±(0.015)	1.6 ±(0.034)	0.00 ±(0.0000)
	Integer PID controller using DE	0.21 ±(0.011)	0.435 ±(0.078)	0.2 ±(0.010)	0.00 ±(0.0000)
	Fractional Controller using PSO	0.12 ± 0.012)	0.982 ±(0.101)	0.1 ±(0.009)	0.0001 ±(0.0000)
	Fractional Controller using GA	0.27 ± 0.017)	1.312 ±(0.313)	0.3 ±(0.013)	0.0522 ±(0.0037)

Table 4. FOPID controller transfer functions as found using the modified DE for two test problems.

Process Plant Transfer Function: $G_p(s)$	Controller Transfer Function $G_c(s)$
$\frac{k}{(Js+b)(Ls+R)+k^2}$ $J=0.01, b=0.1,$ $k=0.01,$ $R=1, L=0.5,$	$36.762+221.852 s^{-0.668}+40.719 s^{0.8}$
$\frac{1}{0.9s^{0.3}+0.6s^{0.8}+1}$ (Hypothetical Plant)	$1.72+41.524 s^{-0.668}+1.59 s^{0.824}$

5.5. Results

Figure 3 shows the dynamic response characteristics of the closed loop systems for design problems I and II, as specified in Table 1. The integer order PID controller as marked in Figure 3 was obtained by minimizing the same objective function (in equation (27)) in three dimensions, taking $\lambda = \mu = 1$. Table 4 provides the FOPID Controller transfer functions for two test problems as found with modified DE. Table 5 reports the maximum overshoot (in %), rise time (in sec) and steady state error (in %) for the unit step response of each closed loop system under the different PID controllers considered here. All entries in Table 5 are the mean of the 25 independent runs of the modified DE, the HPSO-TVAC and the binary GA algorithm and computed the respective standard deviations as well. It is noted that for the given common performance criteria on peak overshoot M_P, rise time t_{rise} sec, and steady state error e_s the fractional order controller achieves better results than its integer counterpart in general. The DE based FOPID controller provides results closest to the three user specifications as listed in Table 1 in each case.

6. CONCLUSIONS

An intelligent optimization method for designing fractional order PID (FOPID) controllers based on the DE is presented in this paper. Fractional calculus can provide novel and higher performance extension for FOPID controllers. However, the difficulties of designing FOPID controllers increase, because FOPID controllers also take into account the derivative order and integral order in comparison with traditional PID controllers. To design the parameters of the FOPID controllers efficiently, the DE/rand/1/bin algorithm is modified with respect to its scale factor *F* and Crossover Rate *CR*. The proposed method has been shown to outperform a state-of-the-art version of the PSO algorithm and a binary GA based method especially for the fractional order plants. The proposed scheme of fractional PID controller design will thus find extensive commercial application in the next generation controller design.

7. REFERENCES

[1] Oldham, K. B. & Spanier, J.: The Fractional Calculus. Academic Press, New York, 1974.

[2] Lubich, C. H.: Discretized fractional calculus. SIAM J. Math.Anal., **17**(3), 704–719, 1986.

[3] Miller, K. S. and Ross, B. An Introduction to the FractionalCalculus and Fractional Differential Equations. Wiley, New York (1993).

[4] Oustaloup, A. Fractional order sinusoidal oscilators: Optimization and their use in highly linear FM modulators. IEEE Transactions on Circuits and Systems, 28(10), 1007–1009. (1981).

[5] Chengbin, Ma and Hori, Y.: The application of fractional order PID controller for robust two-inertia speed control, Proceedings of the 4-th International power electronics and motion control conference, Xi'an, Aug. 2004.

[6] Podlubny, I.: Fractional-order systems and PI□D□ controllers, IEEE Trans. On Automatic Control, vol.44, no. 1, pp. 208-213, 1999.

[7] Chen, Y.Q., Xue, D., and Dou, H.: Fractional calculus and biomimetic control, in: Proceedings of the First IEEE International Conference on Robotics and Biomimetics (RoBio04), Shengyang, China, August 2004, IEEE.

[8] Nakagawa M. and Sorimachi, K. : Basic characteristics of a fractance device, IEICE Trans. Fundamentals, vol. E75-A, no. 12, pp. 1814–1819, 1992.

[9] Y.Q. Chen, H. Ahn, D. Xue, Robust controllability of interval fractional order linear time invariant systems, in: Proceedings of ASME 2005 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, paper# DETC2005-84744, Long Beach, CA, September 24–28, 2005, pp. 1–9.

[10] Vinagre, B. M., Podlubny I., Dorcak L., and Feliu V., On Fractional PID Controllers: A Frequency Domain Approach, Proceedings of IFAC Workshop on Digital Control - PID'00, Terrassa, Spain, 2000

[11] Petras, I, The fractional order controllers: methods for their synthesis and application, J. Electrical Engineering, Vol.50, No.9-10, pp.284-288, 1999.

[12] Dorcak, L., Petras, I., Kostial, I. and Terpak, J., State-space controller design for the fractional-order regulated system, Proceedings of the ICCC'2001, Krynica, pp. 15 - 20. May, 2001.

[13] Astrom, K., Hagglund, T.: PID Controllers; Theory, Design and Tuning., Instrument Society of America, Research Triangle Park, 1995.

[14] Price, K., Storn, R., and Lampinen, J.: Differential Evolution - A Practical Approach to Global Optimization, Springer, ISBN: 3-540-20950-6, 2005.

[15] Storn, R., Price, K.: Differential evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, Journal of Global Optimization, 11(4) (1997) 341–359.

[16] Vesterstrøm, J., Thomson, R.: A Comparative Study of Differential Evolution, Particle Swarm Optimization, and Evolutionary Algorithms on Numerical Benchmark Problems, Proc. Sixth Congress on Evolutionary Computation (CEC-2004). IEEE Press.

[17] Ratnaweera, A, Halgamuge, K, S.: Self organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. IEEE Trans. on Evolutionary Computation (2004) 8(3): 240-254

[18] Holland, J. H.: Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Harbor (1975).

[19] Cao, J , Liang, J, Cao, B.:Optimization of Fractional Order PID Controllers Based on Genetic Algorithms, Proceedings of the International Conference on Machine Learning and Cybernetics,Guangzhou,18-21 August 2005.

[20] Caputo, M.:Linear model of dissipation whose Q is almost frequency independent—II. *Geophys. J. R. Astr. Soc*., vol. 13, pp. 529–539, 1967.

[21] Caputo, M.: *Elasticita e Dissipacione*, Bologna: Zanichelli, 1969.

[22] Price, K.: An Introduction to Differential Evolution in Corne, D, Dorigo, M and Glover, V (eds.), New Ideas in Optimization, Pgs-79-108,McGraw-Hill,UK,1999.

[23] Das, S., Konar, A., Chakraborty, and U.K.: Two improved differential evolution schemes for faster global search, ACM-SIGEVO Proceedings of GECCO, Washington D.C., June 2005, pp. 991-998.

[24] Podlubny, I.: Fractional Differential Equations, Academic Press, San Diego-Boston-New York- London-Tokiyo-Toronto, 1999, 368 Pages.

[25] Stengel, R.F.: Optimal Control and Estimation, Dover Publications, New York, 1994.

[26] DE source codes online : http://www.icsi.berkeley.edu/~storn/code.html