Design of fractional order PID controller for DC motor using evolutionary optimization techniques

ASHU AHUJA*, SANJEEV KUMAR AGGARWAL 1
Electrical Engineering Department,
M. M. Engineering College, Mullana, Ambala, Haryana,
INDIA
ashu.valecha.ahuja@gmail.com, vasusanjeev@yahoo.co.in
1 The corresponding author

Abstract: - In this paper, a comparison of various Evolutionary Optimization (EO) techniques has been performed for controller tuning (PID and FOPID) for DC motor speed control applications. Three techniques have been considered, which are: genetic algorithm (GA), particle swarm optimization (PSO), and differential evolution (DE). A fixed structure PID and FOPID controller has been considered. Usually in literature, comparison has been carried out based on time domain performance indices; whereas, in this work, mixed sensitivity $H_{\infty}$ method is considered as the fitness function for the comparison of various EO techniques in order to assess the robustness of the designed system. It has been shown that GA optimized FOPID controller has better robustness with respect to model uncertainties.

Key-words: Controllers, differential evolution, DC motor, genetic algorithm, mixed sensitivity function, particle swarm optimization

1 Introduction

The proportional–integral–derivative (PID) controller is the most widely used controller for industrial applications due to its simple design and ability to provide effective control [1]. But, many real-world physical systems need fractional calculus as they are modelled by fractional-order differential equations, i.e., equations involving non-integer order integrals and derivatives such as dynamic systems with materials having memory and hereditary effects [2], chaotic systems [3], automatic voltage regulator (AVR) [4], permanent magnet synchronous machine (PMSM) AC servo system [5]. The emergence of ‘fractance’ and memristors [6] and progress in the synthesis of real non-integer differentiator has led to the development of fractional-order controllers, including fractional-order PID controllers (FOPID) [1]. The main advantage of using fractional-order controllers for a linear control system is that the time and frequency responses can be obtained in the form of functions other than that of exponential type and better performance as compared to integer-order controllers can be obtained. In a PID controller, the derivative and the integral order are of integer nature; whereas, in FOPID, the derivative and integral order are of fractional nature rather than integer [2]. The key challenge of designing a FOPID controller is to determine the two key parameters $\lambda$ (integral order) and $\mu$ (derivative order) apart from usual tuning parameters of PID. Both $\lambda$ and $\mu$ are in fraction, which increases the robustness of the system and gives an optimal control [7-9].

DC motor is widely used for industrial applications due to its easy controllability and high performance [10-11]. In DC motor speed control, robustness is the primary requirement of design followed by the performance of the system. Therefore, many controllers are designed to ensure the disturbance rejection and robustness. While many control structures have been proposed in literature for DC motor speed control, such as adaptive variable structure control [12], differential feedback control [13], fuzzy logic DC motor speed control [14]; yet to incorporate robustness in the controller design, $H_{\infty}$ is the most popular optimal control technique [15]. A design procedure which incorporates loop shaping methods to obtain performance and robust stability trade off and a particular $H_{\infty}$ optimization problem to guarantee closed loop stability has been proposed in [16]. Moreover, adaptive neural method [17], robust controller [18] and robust adaptive discrete variable structure control scheme for speed control of DC motor [12] have also been introduced. A multi objective formulation has also been proposed [19].

Numerous controllers have been developed in literature for DC motor speed control namely: PID controller, fuzzy logic controller, neural network [20], fuzzy-neural network, fuzzy-genetic algorithm [21], fuzzy-ant colony, fuzzy-swarm [14], fuzzy-sliding mode control [22]. Many techniques have been used to tune the parameter of these controllers.
such as particle swarm optimization (PSO) [23-24], augmented Lagrangian particle swarm optimization [25], improved differential evolution [7], and FOPID controllers tuned by Ziegler-Nichols type rules [26]. Among these techniques evolutionary optimization (EO) techniques have emerged as popular controller design techniques. However, a comparison of various evolutionary optimization (EO) techniques like genetic algorithm (GA), Particle Swarm Optimization (PSO) and Differential Evolution (DE) for FOPID tuning still needs to be explored and has been investigated in this work.

In this paper, GA, PSO and DE have been employed to design an FOPID controller for DC motor speed control. The proposed FOPID controller is simulated with six tuning parameters \((k_p, k_i, k_D, \tau_D, \lambda, \mu)\) and its performance is compared with optimally tuned PID controller \((k_p, k_i, k_D, \tau_D)\). It has been concluded that with respect to system uncertainties, robustness is significantly improved by FOPID controller. In literature, many authors have explored the FOPID controller tuning using time domain performance indices such as IAE, ISE, ITAE and ITSE [27-28], having constraints on gain crossover frequency, phase margin, robustness to variation in the gain of the plant, robustness to high frequency noise and good output disturbance rejection [29] or minimizing the characteristics equation [30]; but, issues of stability and robustness with mixed sensitivity function involving FOPID controller have not been adequately addressed.

The paper is presented as follows. Sections 2 and 3 present the overview of the concepts of fractional calculus and EO techniques, respectively. Design of controllers for DC motor is described in Section 4. Section 5 is devoted to tuning of the proposed FOPID and PID controllers using various EO techniques like GA, PSO and DE. Section 6 consists of comparison of FOPID and PID controllers. Section 7 concludes the paper.

2 Fractional Calculus

Fractional calculus may be explained as the extension of the concept of a derivative operator from integer order \(n\) to arbitrary order \(v\) where \(v\) may be a real value or a complex value or may be a complex valued function \(v = v(x, t)\):

\[
\frac{d^n}{dx^n} \rightarrow \frac{d^v}{dx^v}
\]

The most commonly referred definition of fractional derivatives and fractional integrals was given by G. F. B. Riemann and J. Liouville, according to which, the fractional integral of order \(v > 0\) for a function \(f(t)\) is given by:

\[
_{a}D_t^{-v} f(x) = \frac{1}{\Gamma(v)} \int_{a}^{x} (x-t)^{v-1} f(t)dt, \quad n \in \mathbb{N}
\]

with the condition that \(f(x)\) and \(_{a}D_t^{-v} f(x)\) are causal functions. For initial conditions to be zero, the Laplace transform of \(D\) is given by \(L\left(D^{-v}\right) = s^{-v} F(s)\) i.e., for zero initial conditions, the system whose dynamic behavior described by differential equations having fractional derivatives results in transfer functions with fractional orders of \(s\) [31-33]. To simulate fractional order of \(s\) in MATLAB, this is to be approximated by usual integer order transfer function having an infinite number of poles and zeroes. Oustaloup proposed a method of approximation of a function of the form [34]:

\[
H(s) = s^\mu, \mu \in \mathbb{R}^+
\]

by a rational function:

\[
H(s) = \frac{1}{C \prod_{i=1}^{N} 1 + \left(\frac{s}{\omega_i}\right)^{\alpha}}
\]

and by applying the following synthesis formulas:

\[
\begin{align*}
\omega_a = & \frac{1}{\alpha} \omega_b; \\
\omega_b = & \frac{1}{\alpha} \omega_a; \\
\frac{\omega_{b+1}}{\omega_b} = & \frac{\omega_{b+1}}{\omega_a} = \alpha \eta > 1
\end{align*}
\]

\[
\frac{\omega_{b+1}}{\omega_b} = \eta > 0; \frac{\omega_{b+1}}{\omega_b} = \alpha > 0; N = \frac{\log(\frac{\omega_b}{\omega_a})}{\log(\alpha)}; \mu = \frac{\log(\alpha)}{\log(\alpha)}
\]

where, \(\omega_a\) is geometrical mean of the unit gain frequency and the central frequency of a band of frequencies distributed around it. That is,

\[
\omega_a = \sqrt{\omega_b \omega_h}
\]

where, \(\omega_b\) and \(\omega_h\) are the high and low transitional frequencies.

3 Evolutionary Optimization (EO) Techniques

Various EO algorithms used in this paper are described as follows:

3.1 Genetic Algorithm (GA)

GA is a global search technique developed by J. Holland in 1970’s. It is used to find true or approximate solutions to complex optimization problems. This technique is inspired by evolutionary biology such as inheritance, mutation, selection and crossover (also called recombination). It is implemented as a simulation problem in which a population of abstract representation (called
chromosome or the genotype or the genome) of candidate solutions (called individual, creatures or phenotypes) moves towards better solution by iteratively applying a set of stochastic operators [35-37].

3.2 Particle Swarm Optimization (PSO)
The PSO was developed by J. Kenedy and R. Eberhart in 1995. It is an evolutionary optimization technique based on the movement and intelligence of swarm that move around in a search space looking for the best solution. Each particle is treated as a point in N-dimensional space which adjusts its flight according to its own flying experience as well as the flying experience of the other particles. Each particle tries to improve its position using its current position, its current velocity, the distance between its current position and its best position ‘pbest’ and the distance between its current position and global best position ‘gbest’. It is an evolutionary technique similar to GA; but, it has a faster convergence [38-40].

3.3 Differential Evolution (DE)
DE was introduced by Storn and Price in 1996. It is a stochastic, population based optimization algorithm like GA. But one big difference is that DE is developed to optimize real parameters or real valued functions that are non-differentiable, non-continuous, non-linear, noisy, flat, multi dimensional or have many local minima. As a result, the idea of mutation and crossover are substantially different in both the techniques. DE has better convergence to global optimum, more accurate and reduced number of simulations in comparison to other optimization techniques [41-42].

4 Controller design for DC motor speed control
A block diagram of speed control system involving DC motor and a controller is shown in Figure 1. Here, input is the reference speed, controller is PID or FOPID, output is speed \( \omega(s) \) and disturbance is due to measuring channel, environment or any kind of noise.

4.1 PID controller
PID control with its three term functionality offers the simplest solution to many real world control problems. A PID controller with four tuning parameters is usually selected:

\[
K(s) = k_p + \frac{k_i}{s} + \frac{k_ds}{\tau_ds + 1}
\]  

(7)

Tuning parameters of the controller are:

\[ p = (k_p, k_i, k_D, \tau_D) \]

Figure 1. Controller structure for DC motor

4.2 FOPID controller
The differential equation of a fractional order PI\(^D^\mu\) controller can be described as:

\[
u(t) = k_p e(t) + k_i \frac{D^\mu}{\tau_D^\mu} e(t) + k_D D^{-\mu} e(t)
\]  

(8)

and its transfer function can be given as:

\[
K(s) = k_p + k_i s^{-\mu} + k_D s^\mu
\]  

(9)

FOPID controller involves selection of five parameters: three parameters (same as PID) and two fractional parameters \( \lambda, \mu \). More flexibility is added in accomplishing control objective by this expansion. A drawback with derivative action is the noise amplification, which can be attenuated by replacing the term \( k_D s^\mu \) by \( \frac{k_D s^\mu}{\tau_D s + 1} \) (insertion of pole in the vicinity of the zero of the derivative action). The transfer function of FOPID controller becomes:

\[
K(s) = k_p + k_i s^{-\mu} + \frac{k_D s^\mu}{\tau_D s + 1}
\]  

(10)

So, there are six parameters to tune now:

\[ p = (k_p, k_i, k_D, \tau_D, \lambda, \mu) \]

4.3 DC motor model
DC machines are characterized by their versatility. By means of various combinations of shunt, series and separately excited field windings, they can be designed to display a wide variety of volt-ampere or speed-torque characteristics for both dynamic and steady state operation and are frequently used in many applications requiring a wide range of motor speeds and a precise output motor control. The schematic diagram of a typical DC motor and its model are shown in Figure 2 and 3 respectively.

From Figure 3, the transfer function from the input voltage, \( V(s) \), to the output velocity, \( \omega(s) \) and to the output angle, \( \theta(s) \) can be written as:

\[
\omega(s) = \frac{K}{(Ls + R)(Js + B) + K^2}
\]  

(11)
\[ \theta(s) = \frac{K}{V(s)} = \frac{1}{s[(Ls + R)(Js + B) + K^2]} \] (12)

where, \( K \) is the emf (Nm/A), \( L \) is inductance (henry), \( R \) is electrical resistance (ohm), \( J \) is the moment of inertia of the rotor (kg.m\(^2\)/s\(^2\)), and \( B \) is the damping ratio of the mechanical system.

The fitness function is based on the concept of robust mixed-sensitivity control which is the infinity norm of weighted sensitivity and complimentary sensitivity function [10]. In this control method, the norm of weighted sensitivity and complimentary sensitivity function \([10]\). In this control method, the magnitude of the complementary sensitivity function \( W_2 \) for the disturbance rejection of the system to limit the performance weighting function, which is specified as:

\[ \omega(s) = \frac{0.1}{V(s)} = \frac{0.1}{0.001s^3 + 0.14s + 0.41} \] (16)

The synthesis procedure of \( H_c \) controller can be done only by selecting proper weight functions. The selection purely depends on the plant model. There are no hard and fast rules for selecting the performance weight function and the robustness weighting function. An iterative work with assumed initial values is usually conducted to find out the weight functions \( W_1 \) and \( W_2 \). The guidelines for selection of these weighting functions have been explained in [43]. Inverse of \( W_1 \) should reflect the desired shape of the sensitivity function \( S \); whereas, inverse of \( W_2 \) should exhibit the shape of the complementary sensitivity function \( T \). The parameters (coefficients and constants) of \( W_1 \) are adjusted such that the singular value curve of \( S \) remains below the singular value curve of inverse of \( W_1 \). Similarly, the parameters of \( W_2 \) are adjusted such that the singular value curve of \( T \) remains below the singular value curve of inverse of \( W_2 \).

In this work, four cases of different weighting functions have been considered in order to make detailed analysis. The frequency dependent weighting functions considered are:

\[ W_1 = \frac{p_1 s^2 + p_2 s + p_3}{q_1 s^2 + q_2 s + q_3} \] and

\[ W_2 = \frac{q_1 s^2 + q_2 s + q_3}{p_1 s^2 + p_2 s + p_3} \]
In Case 1, \( W_1 \) and \( W_2 \) have been taken from [10]. In Case 2, \( W_2 \) has been taken from [10] and \( W_1 \) has been tuned using EO technique. In Case 3, \( W_2 \) is taken from [10] and denominator of \( W_1 \) is tuned using EO technique. In Case 4, \( W_2 \) is taken from [10] and numerator of \( W_1 \) is tuned using EO technique.

The tuning of controller parameters in Case 1 is explained below:

### 5.1 Tuning using GA (Case 1)

In Case 1, the EO algorithm (GA, PSO, DE) aims to find optimal value of FOPID controller \([K_p, K_i, K_d, \tau_d, \lambda, \mu]\) to minimize the objective function as given in (13). The initial value, lower and upper bound of solution variables are set at \([92.38, 198.93, 7.24, 0.0006, 0, 0]\); \([10, 100, 1, 0.0001, 0, 0]\) and \([1000, 1000, 100, 0.1, 1, 1]\) respectively. The upper and lower bounds of solution variables have been selected after performing a number of experiments. It was observed that if the range of the bounds is increased then the search space increases and this may lead to non-convergence of the optimization algorithm. On the other hand, if the range is reduced then sufficient optimization is not achieved.

For the proposed simulation, the size of population is taken as 20 and number of generations is equal to 100. The mutation rate is chosen to be 0.05. The GA gets converged in 56 generations with the optimal solution, \([291.4001, 672.8005, 99.9982, 0.0832, 0.9969, 1.0000]\), which on substitution in (10) provide following controller \(K(s)\):

\[
K(s) = \frac{291.4001 + 672.8005}{s^{0.9969} + 99.9982^{1.0000} + 0.0832s + 1}
\]

According to the theory of robust mixed sensitivity control, the fitness function chosen should be less than 1 and the evaluated infinity norm is 0.5361. Therefore, designed system is robust.

The same GA specifications are used to tune PID controller parameters \([K_p, k_i, k_P, \tau_P]\). The initial value, lower and upper bound of solution variable are set at \([95.38, 200.93, 10.24, 0.0009]\), \([1, 10, 1, 0.0001]\) and \([100, 1000, 100, 0.1]\) respectively. The GA converges in 54 generations with the optimal solution, \([29.4099, 825.4733, 9.5151, 0.0158]\); which on substitution in (7) provide following controller \(K(s)\):

\[
K(s) = \frac{29.4099 + 825.4733}{s + 9.5151 + 0.0158s + 1}
\]

The system obtained by the estimated controller is robust as the fitness function obtained is 0.7221, which is less than 1. Figure 4 shows the sensitivity function of the plant using PID and FOPID controller. It is shown that with FOPID, disturbance gets attenuated in larger frequency range. Figure 5 shows that the noise attenuation is more with the FOPID controller. Figure 6 gives the step response of the plant with PID and FOPID controller. A step disturbance is introduced at \(t=3.5\) sec. and it is observed that steady state value is obtained far earlier with FOPID than PID controller.
5.2 Tuning using PSO (Case 1)

Initialization parameters used for PSO are: population size = 25, maximum number of iterations = 2500, minimum and maximum velocities are 0 and 2, cognitive acceleration coefficient ‘C₁’ = 2.1, social acceleration coefficient ‘C₂’ = 1.3, minimum inertia weight = 0.6 and maximum = 0.9. The PSO converges in 34 iterations with the optimal solution of the FOPID controller, [241.4382, 354.8308, 90.8583, 0.0867, 0.9954, 0.9996], which on substitution in (10) provides the following controller $K(s)$:

$$K(s) = 241.4382 + \frac{354.8308}{s^{0.9996}} + \frac{90.8583}{0.0867s + 1}$$

The infinity norm obtained by the evaluated controller is 0.5025, which is less than 1. Consequently, the system is robust.

The same PSO specifications are used to tune PID controller parameters. The initial value, lower and upper bound of solution variable are set at as that in case of GA. The PSO converges in 60 iterations with the optimal solution, [72.1678, 202.3394, 4.2014, 0.0001], which on substitution in (7) provides the following controller $K(s)$:

$$K(s) = 72.1678 + \frac{202.3394}{s} + \frac{4.2014s}{0.0001s + 1}$$

The system obtained by the estimated controller is also robust as the fitness function obtained is 0.5008. Figure 7 shows the sensitivity function of the plant using PID and FOPID controller. It is shown that with FOPID, disturbance is getting attenuated in larger frequency range. Figure 8 shows that the noise attenuation is more with FOPID controller. Figure 9 gives the step response of the plant with PID and FOPID controller.

5.3 Tuning using DE (Case 1)

Initialization parameters used for DE are: value to reach (VTR) = 0.5 (DE will stop its minimization if either the best parameter vector has found a value <= “VTR” or number of iterations reached its
The similar exercise was carried out in Case 2, Case 3, and Case 4 as well; where, in addition to the controller parameters, coefficients of weighting function $W_j$ were also optimized by various optimization techniques. The optimized parameters values of FOPID and PID controllers are given in Table 2-5 for Case 1, Case 2, Case 3 and Case 4 respectively. It can be observed that optimized parameters values for both the controllers using GA and PSO are close to each other as compared to that obtained using DE.

Here, infinity norm comes out to be 0.5009; therefore, the system is robust. The same DE specifications are used to tune PID controller parameters. The initial value, lower and upper bound of initial values of solution variable are set at as in earlier optimization techniques. The DE in 200 iterations converges with the optimal solution, $[65.5010, 985.1620, 12.1832, 0.00001]$, which on substitution in (10) provides the following controller $K(s)$:

$$K(s) = 125.8399 + \frac{82.2894}{s^{1.1018}} + \frac{52.4687}{0.1972s + 1} + \frac{985.1620}{s} + \frac{12.1832}{0.0001s + 1}$$

Here, infinity norm comes out to be 0.5005; therefore, the system is robust. The Figure 10 shows the sensitivity function of the plant using PID and FOPID controller. It shows that with FOPID disturbance is getting attenuated in larger frequency range. Figure 11 shows that the noise attenuation is more with the FOPID controller. Figure 12 gives the step response of the plant with PID and FOPID controller.

![Figure 10. Sensitivity function of DC motor with PID and FOPID controller using DE](image)

**Table 2. Optimized parameters values for Case 1**

<table>
<thead>
<tr>
<th>Optimized Parameters</th>
<th>Value of Optimized parameters and fitness function(f) with FOPID controller</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k_p$</td>
<td>GA</td>
</tr>
<tr>
<td>$k_i$</td>
<td>FOPID</td>
</tr>
<tr>
<td>$k_D$</td>
<td>99.998</td>
</tr>
<tr>
<td>$\tau_D$</td>
<td>0.0832</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>9.5151</td>
</tr>
<tr>
<td>$\mu$</td>
<td>1.000</td>
</tr>
<tr>
<td>$f$</td>
<td>0.5361</td>
</tr>
</tbody>
</table>

Assumed parameters: $p_1=0$, $p_2=0.5$, $p_3=10$, $q_1=0$, $q_2=1$ and $q_3=0.001$; * Not applicable

**Table 3. Optimized parameters values for Case 2**

<table>
<thead>
<tr>
<th>Optimized Parameters</th>
<th>Value of Optimized parameters and fitness function(f) with FOPID controller</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k_p$</td>
<td>GA</td>
</tr>
<tr>
<td>$k_i$</td>
<td>FOPID</td>
</tr>
<tr>
<td>$k_D$</td>
<td>424.54</td>
</tr>
<tr>
<td>$\tau_D$</td>
<td>0.0819</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>9.5151</td>
</tr>
<tr>
<td>$\mu$</td>
<td>1.000</td>
</tr>
<tr>
<td>$f$</td>
<td>0.5361</td>
</tr>
</tbody>
</table>

Assumed parameters: $q_1=1$; * Not applicable
6 Comparison of PID and FOPID controlled process

The comparison drawn between the two controllers has been shown in Table 6 to Table 8 for GA, PSO and DE respectively for all the four cases.

From Table 6-8, it can be observed that the rise time, settling time and peak overshoot are reduced (improved) in case of a FOPID controller than that in the case of a PID controller; but, in case of PSO and GA; overshoot is almost negligible when PID controller is applied. It can also be observed that percentage reduction in performance characteristics is more in case of GA. The variation in the value of the fitness function during iterative process of GA has been shown in Figure 13 and Figure 14 for PID and FOPID respectively. The variation in the value of the fitness function during iterative process of PSO has been shown in Figure 15 and Figure 16 in case of PSO for PID and FOPID respectively. The variation in the value of the fitness function during iterative process of DE has been shown in Figure 17 and Figure 18 for PID and FOPID respectively. It is observed that though PSO requires less number of iterations than the other techniques but GA converges more sharply than the other techniques i.e. variation in the fitness function with successive iteration is very less in GA in comparison to other optimization techniques. Further, it can also be observed that there are many large spikes in fitness function of PID controller in Figure 15 (PSO) and Figure 17 (DE); whereas; in case of Figure 13 (GA) the convergence is smooth. This may be due to the characteristic of the process undertaken by different algorithms for searching the search space.

7 Conclusion

In this paper, a comparison of various evolutionary optimization techniques for the designing of PID and FOPID controllers for DC motor speed control has been investigated. The comparison involves frequency domain performance criterion. In order to examine the effect of cost function, different cases are examined by taking different coefficients of performance weighting function unknown. The performance of PID and FOPID is compared using GA, PSO and DE for different cases and it is observed that in each case, FOPID has better performance characteristics in terms of rise time, settling time and overshoot. Considering all the results, it is also observed that though PSO requires less number of iterations than the other techniques but variation in the fitness function with successive iteration is very less in GA in comparison to other optimization techniques.

### Table 4. Optimized parameters values for Case 3

<table>
<thead>
<tr>
<th>Optimized Parameters</th>
<th>Value of Optimized parameters and fitness function(f) with FOPID controller</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>GA PID FOPID PID FOPID PID FOPID PID</td>
</tr>
<tr>
<td>$k_i$</td>
<td>96.269 34.571 10.160 42.999 363.87 24.969</td>
</tr>
<tr>
<td>$k_i$</td>
<td>134.65 65.504 100.13 73.691 185.86 100.12</td>
</tr>
<tr>
<td>$k_p$</td>
<td>80.457 3.2674 45.898 4.8694 985.13 9.7017</td>
</tr>
<tr>
<td>$A_o$</td>
<td>0.4902 0.0137 0.0577 0.0005 0.8464 0.8607</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>0.9702 * 0.1871 * 0.7679 *</td>
</tr>
<tr>
<td>$\mu$</td>
<td>0.9902 * 1 * 0.7478 *</td>
</tr>
<tr>
<td>$q_1$</td>
<td>0.8547 9.9966 0.8765 9.9892 1.6567 4.4505</td>
</tr>
<tr>
<td>$q_2$</td>
<td>0.8135 9.3605 0.8846 9.9352 0.8048 7.6576</td>
</tr>
<tr>
<td>$f$</td>
<td>0.5880 0.1892 0.5721 0.1854 0.3664 0.4173</td>
</tr>
</tbody>
</table>

Assumed parameters: $p_1=0$, $p_2=0.5$, $p_3=10$, $q_1=0$

* Not applicable

### Table 5. Optimized parameters values for Case 4

<table>
<thead>
<tr>
<th>Optimized Parameters</th>
<th>Value of Optimized parameters and fitness function(f) with FOPID controller</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>GA PID FOPID PID FOPID PID FOPID PID</td>
</tr>
<tr>
<td>$k_i$</td>
<td>62.919 34.571 67.025 42.999 180.40 24.969</td>
</tr>
<tr>
<td>$k_i$</td>
<td>280.51 65.504 100.13 73.691 1736.6 100.12</td>
</tr>
<tr>
<td>$k_p$</td>
<td>98.186 3.2674 99.936 4.8694 254.00 9.7017</td>
</tr>
<tr>
<td>$A_o$</td>
<td>0.7983 0.0137 0.1569 0.0005 0.9000 0.8607</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>0.5699 * 0.7288 * 0.9000 *</td>
</tr>
<tr>
<td>$\mu$</td>
<td>0.9922 * 0.9994 * 0.9000 *</td>
</tr>
<tr>
<td>$p_1$</td>
<td>0.2343 9.9966 0.2235 9.9892 0.2000 4.4505</td>
</tr>
<tr>
<td>$p_1$</td>
<td>0.4242 9.3605 0.3451 9.9352 5.3000 7.6576</td>
</tr>
<tr>
<td>$f$</td>
<td>0.3009 0.1892 0.2317 0.1854 0.3660 0.4173</td>
</tr>
</tbody>
</table>

Assumed parameters: $p_1=0$, $q_1=0$, $p_3=1$ and $q_2=0.001$

* Not applicable

Figure 12. Closed loop step response with disturbance at t=3.5sec using DE.
### Table 6. Comparison of PID and FOPID controller Using GA

<table>
<thead>
<tr>
<th>Parameters</th>
<th>PID controlled process</th>
<th>FOPID controlled process</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Case1</td>
<td>Case2</td>
</tr>
<tr>
<td>Rise time (s)</td>
<td>0.057</td>
<td>0.221</td>
</tr>
<tr>
<td>Settling time (s)</td>
<td>0.982</td>
<td>2.35</td>
</tr>
<tr>
<td>Peak Overshoot (%)</td>
<td>18.07</td>
<td>4.92</td>
</tr>
<tr>
<td>Steady state error</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

### Table 7. Comparison of PID and FOPID controller using PSO

<table>
<thead>
<tr>
<th>Parameters</th>
<th>PID controlled process</th>
<th>FOPID controlled process</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Case1</td>
<td>Case2</td>
</tr>
<tr>
<td>Rise time (s)</td>
<td>0.045</td>
<td>0.182</td>
</tr>
<tr>
<td>Settling time (s)</td>
<td>0.0775</td>
<td>0.543</td>
</tr>
<tr>
<td>Peak Overshoot (%)</td>
<td>0.881</td>
<td>0</td>
</tr>
<tr>
<td>Steady state error</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

### Table 8. Comparison of PID and FOPID controller using DE

<table>
<thead>
<tr>
<th>Parameters</th>
<th>PID controlled process</th>
<th>FOPID controlled process</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Case1</td>
<td>Case2</td>
</tr>
<tr>
<td>Rise time (s)</td>
<td>0.0197</td>
<td>0.421</td>
</tr>
<tr>
<td>Settling time (s)</td>
<td>0.081</td>
<td>0.239</td>
</tr>
<tr>
<td>Peak Overshoot (%)</td>
<td>4.52</td>
<td>16.6</td>
</tr>
<tr>
<td>Steady state error</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 13. Fitness function using GA for PID controller

Figure 14. Fitness function using GA for FOPID controller

Figure 15. Fitness function using PSO for PID controller

Figure 16. Fitness function using PSO for FOPID controller
References:


[38] M. Clerc, Particle Swarm Optimization, ISTE Ltd., 2006.


