Design of intelligent PID/PI$^\lambda$D$^\mu$ speed controller for chopper fed DC motor drive using opposition based artificial bee colony algorithm

Anguluri Rajasekhar, Ravi Kumar Jatoth, Ajith Abraham

Department of Electrical Engineering, National Institute of Technology, Warangal 506004, India

Abstract

This paper deals with the design, implementation and analysis of an integer order (IO) and fractional order (FO) based Proportional Integral Derivative (PID) controller, for speed regulation in a chopper fed Direct Current (DC) motor drive. The interdependent parameters of PID and FOPID controllers are designed in both the time and frequency domain. In both domains, designs of controllers are formulated as a single objective optimization problem based on time indices integrals and frequency domain rules. Time domain based design of controllers focuses mainly on minimization of indices like rise time, settling time etc. On the other hand frequency domain based design of controllers focuses on achieving iso-damping characteristics, which tries to meet user specified gain crossover frequency or phase margin while also maintaining constant overshoot for wide range of motor gain apart from maintaining optimum time indices. A newly evolved artificial bee colony (ABC) algorithm enhanced with opposition search has been used to perform the optimization task. This type of multiple designs enables users to choose the controller based on their requirement. A comparative study has been made to highlight the advantage of using fractional order controller over conventional integer order PID control scheme for speed regulation. To illustrate the efficacy of the opposition based ABC, we also compared the performance with the conventional ABC, Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) methods. Computer simulations and extensive analysis over results obtained shows the effectiveness of the proposed approach.

1. Introduction

Due to the increase in intricacies of practical applications, industries are in need of higher performance drives, which should be robust and reliable than the existing, apart from satisfying the industrial requirements. A few applications of high-performance drives include areas of rolling mills, machine tools, traction and robotics (Paresh, 1990). These applications need drive systems that can provide quick dynamic response, control characteristics that are insensitive to parameters and swift recovery from speed drops caused via sudden loads (Dubey, 2009). To match these kinds of industrial requirements a high performance drive systems should at least track dynamic speed command and also should maintain load-regulating response. The above requirements can be easily met by employing Direct Current (DC) motor drives (Dubey, 2009). DC motor drives are highly preferred and widely used in applications demanding adjustable speed, frequent starting and good speed regulation because of their excellent control of speed for acceleration and deceleration (Dubey, 2009; Subrahmanyam, 2011). Although AC motor drives popularity is on upswing for many industrial applications, even now the variable speed applications are dominated by DC drives because of their lower cost, simple control and good reliability (Paresh, 1990). The decoupled nature of field magnetomotive force (mmf) and armature mmf is one of main advantage of using DC motors. The power supply of DC motor is directly fed to field of motor, which allows for a precise voltage control, and which in turn aids speed and torque control applications. Another advantage of these motors includes, providing starting and accelerating torques in excess of 400% load and one of the primary applications of DC motors is Electrical traction (Dubey, 2009; Subrahmanyam, 2011).

In control systems perspective, DC motors can be considered as Single Input Single Output (SISO) systems having speed (torque) characteristics well suited with most mechanical loads. This
property of DC motors makes them controllable over wide range of speed by providing good adjustment schemes to terminal voltage. These exemplary features makes DC motors to be a good choice for the state-of-art control algorithms and also the speed control concept of these motors can be extendable to other types of motor as well (Sundareswaran, 2000).

Typically for speed control of DC motor drive a closed loop operation with PI/PID controllers with an inner current and outer speed loop is preferred and employed (Dubey, 2009). Out of various closed loop controllers available, Proportional-Integral-Derivative (PID) based control scheme is widely preferred in many industrial applications because of their ease in structure and implementation. PID based speed control scheme has many advantages like good steady state accuracy, less settling time, quick control and low cost (Subrahmanym, 2011). Conventional notion of integer order (IO) PID controllers has been generalized to a more flexible structure P^{λ}D^{μ} controller by Podlubny (1999) with the fractional differo-integral operators and controller gains as design variables. P^{λ}D^{μ} or fractional order (FO) PID controller that exhibits dynamical behavior is described by differential equations including derivatives and integrals whose order is not an integer (Das, 2011). Literature studies since early 2000s reveal that there has been surge of applications centered on fractional or non-integer calculus and also its extensions to wide gamut of problems ranging from physics to biology (Tenneiro Machado et al., 2010). FO controllers are shown to be more robust and can outperform normal PID controllers if they can be designed effectively (Pan and Das, 2013; Monje et al., 2010).

A typical FOPID controller consists of five interdependent parameters i.e., proportional (K_p), integral (K_i), derivative (K_d), order of derivative (μ) and order of Integral (λ). A FOPID controller can also be used as IOPID controller for λ=μ=1. Hence, designing an optimum PID and FOPID control structures for meeting design specifications of DC drive requires fine tuning of parametric gains (K_p, K_i, K_d, λ, μ). This in return calls for real parameter optimization in 3-dimensional (for PID controller) and five-dimensional hyperspace (for FOPID controller). The design of controllers for application concerned is generally carried out using time and frequency domain approach. The time domain based design of controllers focus on obtaining good response of system while monitoring time indices in transient and steady state. While on other hand with the help of frequency domain rules iso-damping property (Das, 2011) can be introduced into the system. The basic idea of frequency domain approach is to design a controller such that open loop transfer function (T/F) refers to Bode’s ideal T/F which has a property of no variation in phase margin with respect to variation in gain crossover frequency. This kind of design and analysis enables user to choose his desired method. In both domains, the controller variables are related to a single objective function and a global optimization procedure is applied to obtain optimal controller gains.

Out of all existing swarm intelligent algorithms based on bees like BCO, BA, MBO, etc., artificial bee colony (ABC) algorithm has gained wide reputation in swarm intelligence (SI) research community because of its outstanding performance on various kinds of practical problems (Karaboga and Akay, 2009). ABC was first devised and developed by Karaboga (2005) for numerical function optimization. Due to its ease in structure, implementation followed by a well-organized exploitation and exploration mechanisms; ABC remained to be the best contender for existing state-of-art evolutionary methods. A comprehensive view of recent developments and real world applications can be found in Karaboga et al. (2012). To further enhance the performance of conventional ABC algorithm we propose opposition based artificial bee colony (OABC) algorithm, which draws its inspiration from opposition based DE (Rahnamayan et al., 2008). Opposition methods play a pivotal role in discovering global solution when there is a strict limitation on number of iterations. In this present work ABC and OABC are chosen as optimization algorithms for finding the optimal parametric gains of PID and FOPID controller. Computer simulations and wide range of analysis revealed that an optimally designed FOPID controller could outperform PID based control scheme. To check the superiority of OABC on controller design application we also compared its performance with existing meta-heuristics like genetic algorithm (GA), particle swarm optimization (PSO) along with basic version of ABC. Further observations from our studies revealed that both the controllers are dependent upon objective function defined; hence a well-posed objective function, aided by robust optimization technique leads to better performance of plant/process considered.

The rest of paper is organized as follows. Section 2 deals with the literature survey of various design methods used for designing FOPID controllers in time and frequency based domain. In Section 3 mathematical modeling of closed loop control of DC motor drive is performed. A brief introduction to fractional order systems and controllers are provided in Section 4. Rudiments of artificial bee colony and Opposition-ABC algorithms are presented in Section 5. Time domain based and frequency domain based design of controllers and their performances are discussed in Sections 6 and 7 respectively. Conclusions inferred from the results obtained and future work details are provided in Section 8.

2. Related research for designing FO controllers

Among existing quantitative measures of the performance of a system, time integral performance indices are commonly used to evaluate performance of control systems designed (Tavazoie, 2010). The most common indices include ITAE, IAE, ITSE and ISE (Tavazoie, 2010). Several intelligent techniques have been proposed for tuning optimal FOPID controllers based on minimizing these indices (and also variants as well) (Pan and Das, 2013). Rajasekhar et al. (2013) used ITAE based scheme for designing FOPID controller in a PMSM drive using Hybrid Differential Artificial Bee Colony Algorithm. Maiti et al. (2012) also tuned a FOPID controller with help of aforementioned criterion for servo hydraulic positioning system using micro ABC algorithm. Notable applications of ITAE based fractional controllers include works carried out by Cai et al. (2009) using multi-parent evolutionary algorithm, Luo and Li (2009) used blend of Bacterial Foraging and PSO (BF-PSO) approach; load frequency control via fractional controllers by Alomoush (2010); Xue and Chen verified the advantage of both ITAE and ISE in the presence of nonlinearities using DC motor as test bed, for position systems by Chen (Xue et al., 2006). Few interesting approaches based on rest of indices include fractional control of heat diffusion systems by Jesus and Tenneiro Machado (2008). Recently few researchers, apart from considering aforementioned error centered metrics also focused on minimizing control signal. The most widely used one are Integral Squared Controller Output (ISCO), and deviation output (ISDCO) (Pan and Das, 2013). Cao et al. (2005) used GA to optimize weighted sum of ISE and ISCO to find out controller parameters and also Cao and Cao (2005) used PSO for similar application. Pioneering works involving combinations of time integrals and controller integrals and their tuning using various evolutionary algorithms for applications ranging from process control, MIMO systems, networked control systems, chaotic systems can be found in Pan and Das (2013). Apart from single objective optimization, a multi-objective chaotic NSGA-II has been used to handle objectives ITSE and ISCO in AVR systems by Pan and Das (2012). Apart from evolutionary and swarm intelligent based designs, fuzzy rules are also incorporated in FOPID
3. Mathematical modeling of DC motor drive

The speed of separately excited DC motor can be regulated above and below rated speed using chopper as a converter. This paper deals with an armature voltage controlled scheme while maintaining field flux constant. The basic idea of this type of DC motor speed control is that the output speed can be altered by controlling armature voltage for achieving desired speed (under constant field current) (Subrahmaniam, 2011; Gopakumar). To have good speed regulation characteristics, closed loop speed control is preferred. Fig. 1 portrays the actual closed loop speed and current control model of DC motor drive. Block diagram represented in Fig. 1 can be reduced to the model in Fig. 2, involving only single closed speed control loop, mainly used for analysis purposes.

The error signal i.e., resultant of reference speed ($N_{ref}$) and feedback ($N_f$) is fed as input to speed controller. The controller provides required voltage signal ($E_c$) to control the operation of duty cycle of converter. As a result converter (chopper) provides sufficient armature voltage ($V_a$) to bring motor speed to the desired level. The output speed of motor is measured with the help of Tacho-Generator. Since the Tacho voltage obtained will have few ripples in the measured speed (not be in perfect DC form) a filter with an optimal gain is provided in the feedback path of speed control loop (Gopakumar).

3.1. Modeling of separately excited DC motor

A separately excited DC Motor mainly consists of field and armature winding, each of them are provided with separate supply (Fig. 3). Field windings of motor are used to excite the flux. The rotor draws armature current with the help of brush and commutator segment. A separately excited DC motor is excited by a field current ($I_f$) and as a consequence an armature current ($I_a$) flows in the circuit. As a result motor develops a back EMF and a torque to balance the load torque at a particular speed level.

Applying Kirchhoff Voltage Law (KVL) to the equivalent circuit of separately excited DC motor in Fig. 3 will leads to armature voltage Eq. (3.1) and the equivalent Torque generated is given by Eq. (3.2).

$$V_a = I_aR_a + L_a\frac{dl_a}{dt} + E_b$$

(3.1)

$$T_d = J\frac{d\omega}{dt} + B\omega + T_L$$

(3.2)

where $V_a$, armature voltage ($V$); $E_b$, motor back EMF ($V$); $I_a$, armature current ($A$); $R_a$, armature resistance ($\Omega$); $L_a$, armature inductance ($H$); $T_d$, load torque ($N\text{m}$); $J$, moment of inertia ($kg\text{m}^2$); $B$, friction coefficient of motor; $\omega$, angular velocity ($\text{rad/s}$).

Eq. (3.2) can be further simplified by assuming negligible friction in motor ($B=0$). Further denoting $\Phi$ as field flux and $K$ ($V/A\text{s(rad)}$) as back EMF constant, corresponding simplified equations of back EMF and torque developed can be obtained:

$$T_d = J\frac{d\omega}{dt} + T_L$$

(3.3)

$$E_b = K\Phi\omega$$

(3.4)

$$T_d = K\Phi I_a$$

(3.5)

With the help of above equations and by applying Laplace transform to Eq. (3.1) the following equations in “s-domain” can be derived:

$$I_a(s) = \frac{V_a - E_b}{R_a + L_a s} = \frac{V_a - K\Phi\omega}{R_a (1 + L_a/R_a s)}$$

(3.6)
where \( T_a \) represents armature time constant and is given by \( T_a = I_a/R_a \). The equivalent model of DC motor is shown in Fig. 4 (annotated in green). By applying block reduction technique to the motor model in Fig. 4, the following transfer function can be obtained:

\[
\frac{\omega(s)}{V_a(s)} = \frac{1}{1 + K_m} \quad (3.8)
\]

Assuming \( T_m = J R_a/K_m^2 \) as electromechanical time constant and replacing \( K \Phi \) by \( K_m \) \((T_c = 0)\) Eq. (3.8) can be further reduced to Eq. (3.9):

\[
\frac{\omega(s)}{V_a(s)} = \frac{1}{1 + K_m} \quad (3.9)
\]

From Eq. (3.9) it is clear that the two-time constants i.e., \( T_m \) and \( T_a \) of the above system transfer function \( T/F \) are accountable for the response of system.

3.2. Current control loop

Due to the presence of electromechanical time constant, motor will consume some time to speed up (Gopakumar). On the other hand speed controller used will be acting very fast. Initially speed feedback is zero, and this result in maximum converter voltage signal \( V_a \). This ultimately leads to large amount of current flow in the motor because of zero back EMF. In course of time this current may exceed the motor maximum current limit and can damage the motor windings. Hence a need to control motor armature current. This problem can be eliminated if closed loop current control scheme can be implemented. As it is clear that speed error indicates the required acceleration, more torque is required for desired speed, which is nothing but more current requirement.

Whenever there is an increase in reference current \( I_{ref} \) as signaled by speed controller, motor armature current increases and thereby providing required acceleration and torque. On the other hand if speed error reduces, the current and torque reduces to drive motor to desired speed. The main objective of current control loop is to make the armature current follow the current reference signal \( I_{ref} \). As the components such as power electronic converter, motor windings etc., are designed for particular rating care has to be taken such that the current or voltage signals higher than these rating should be clamped out. Hence current control loop acts as a heart of DC motor drive in providing safety for the equipment apart from assisting in getting desired response.

Due to this type of control, applied voltage \( V_a \) not only depends on speed error but also relies on current error as well. Hence a current controller is required in the closed loop current control to ensure that \( V_a \) applied during positive and negative torques will not draw more than the rated current. In this work we considered chopper as converter. Typically chopper is a static power electronic component, which converts fixed DC input voltage to a variable DC output. Since they involve one-stage conversion choppers are more efficient when compared to rest of them. Since, chopper works on the principle of pulse width modulation it involves no time delay in its operation. Hence, a chopper can be represented by a simple constant gain \( K_c \). Detailed explanation and operation of DC-choppers and their relevant mathematical expressions can be obtained in Bimbhra (2006).

Since there will be more amount of current flow in armature circuit of motor, the current controller should be designed for extreme conditions (i.e., back EMF = 0). Equivalent model of current control loop can be found in Fig. 4 (annotated in red). The equivalent Transfer function of the close loop current control is given by the following equation:

\[
I_{a}(s) \left( \frac{1}{V_a(s)} \right) = \frac{(K_i K_c/R_a) s (1+T_c)}{1+(K_i K_c/R_a) s (1+T_c) s (1+T_a) s (1+T_c)} \quad (3.10)
\]

To achieve faster response the current controller time constant \( T_c \) has to be chosen such that largest time constant in above transfer function \( T/F \) should be canceled. This is achieved by assuming \( T_c = T_a \); now Eq. (3.10) can be reduced to Eq. (3.11):

\[
I_{a}(s) \left( \frac{1}{V_a(s)} \right) = \frac{(K_i K_c/R_a) s (1+T_c)}{1+(K_i K_c/R_a) s (1+T_c) s (1+T_a) s (1+T_c) (1+T_a)} \quad (3.11)
\]

where \( K_0 = (K_c K_i)/(R_a T_a) \); \( K_c = \) current controller gain constant; \( K_i = \) current loop filter lag factor; \( T_a = \) current loop filter lag. The characteristic equation of Eq. (3.11) will be equivalent to standard second order system equation. To get proper response, the constant \( \xi \) of the 2nd order system is set to 0.707 and equating this to our current \( T/F \) (3.11) results in value of \( K_0 \) in terms of \( K_2 \) and is given by \( K_0 = 1/(2K_2 T_a) \).

From Eq. (3.11) it is very clear that zero in the T/F may result in overshoot. Hence a time lag filter is used to cancel its effect. As current loop time constant is much higher than the filter time constant, a small time delay will not affect the system.

**Fig. 4.** A complete block diagram representation of speed control of DC motor drive.
performance. By above expressions Eq. (3.11) is simplified to following form:

\[
\frac{I_a(s)}{I_{aT}(s)} = \frac{1/K_2}{2T_s^2 + 2T_s + 1}
\]  

(3.12)

3.3. Speed control loop

After considering few approximations in the aforementioned equations, complete layout of closed loop speed control scheme involving inner current control loop in chopper fed DC motor drive is shown in Fig. 4. The speed controller either PID or FOPID has to be designed based on user/drive specifications such that desired speed is achieved within the constraints levied. The controller is designed in two different environments i.e., using time domain indices and using frequency domain rules. As the time domain based approach requires minimization of objective function based on time integrals, closed loop control demonstrated in Fig. 4 will be sufficient. On the other hand frequency domain based design of controllers requires frequency responses for DC drives and controllers. Hence the individual blocks of model depicted in Fig. 4 are reduced to convenient form based on discussions in Section 3. A reduced form of speed controller loop is depicted in Fig. 5, which is a simplified model of Fig. 4.

4. Fundamentals of fractional order controllers

Since the inception of fractional order PID or PI^D^μ controllers (Podlubny, 1999), they are used in diverse fields of engineering. They are also known as generalized version of conventional PID controller. The basic idea of fractional order controllers derives its origin from the concept of fractional calculus (Das, 2011; Oldham and Spanier, 1974). Fractional calculus is a branch of mathematical analysis (having 300 years history) that considers and studies the possibility of taking arbitrary real number (not necessarily real) power of the integration and differential operator. Although most popular definitions of fractional derivative like Riemann Liouville and Grunwald–Letnikov are prevalent in fractional calculus literature, Caputo definition is widely preferred for designing fractional order systems (Das, 2011). The main advantage of using Caputo’s definition is that it allows for consideration of easily construable bounded initial conditions, which means the derivative of constant is equal to zero. This particular definition is used to derive fractional order T/F models from fractional order differential equations with zero initial conditions. The n-th order derivative of a function f(t) with respect to time t according to Caputo definition is given as follows (Das, 2011):

\[
D^n f(t) = \frac{1}{\Gamma(n - \alpha)} \int_0^t \frac{D^{n-\alpha} f(t)}{(t - \tau)^{n-\alpha}} d\tau \quad \alpha \in \mathbb{R}^+, \quad m - 1 \leq \alpha < m,
\]

(4.1)

Applying Laplace transformation to Eq. (4.1) the following s domain representation can be obtained:

\[
\int_0^\infty e^{-s\tau} f(t) \, dt = \frac{1}{s^n} F(s) - \sum_{k=0}^{n-1} s^{n-k-1} D^k f(0)
\]

(4.2)

where

\[
\Gamma(n) = \int_0^\infty e^{-\tau} \tau^{n-1} d\tau \text{ is the Gamma function}
\]

\[
F(s) = \int_0^\infty e^{-sf(t)} \, dt \text{ is Laplace transform of } f(t)
\]

To obtain equivalent transfer function representation of the linear FO system, it is considered that initial conditions of FO differential equation are zero. With these assumptions the time domain operator \( D^n \) can be simply represented in frequency domain as \( s^n \). To perform a fractional integral operation \( \alpha \) operator is multiplied by negative sign. Hence by above expressions FOPID controller can be represented as weighted sum of fractional operators along with controller gains. The continuous transfer function representation of FOPID controller is given by (Podlubny, 1999; Pan and Das, 2013):

\[
C(s) = K_p + \frac{K_I}{s^\alpha} + K_D s^\mu
\]

(4.3)

The output response or control signal provided by PI^D^μ controller in time domain may be given as follows:

\[
u(t) = K_p \epsilon(t) + K_I D^{-\alpha} \epsilon(t) + K_D s^\mu \epsilon(t)
\]

(4.4)

where \( \epsilon(t) \) represents error signal and \( u(t) \) indicates the control signal produced. Hence a typical controller consists of proportional gain \( K_p \), proportional gain \( K_I \), derivative gain \( K_D \) and two more fractional operators i.e., integral order \( \lambda \) and derivative order \( \mu \). When \( \lambda = 0 \) and \( \mu = 0 \) it simplifies to classical integral (I) control, for \( \lambda = 0 \) and \( \mu = 1 \) we get normal PD control, \( \lambda = 1 \) and \( \mu = 0 \) implies PI controller and for \( \lambda = 1 \) and \( \mu = 1 \) we get classical PID controller in parallel structure. This justifies that all these classical types of PID controllers are the special cases of FOPID controller. The FOPID controller generalizes the integer order PID controller and also from Fig. 6 it can be perceived that it expands from point representation to plane. This extension provides good supervisory control over the plant/process considered.

In this paper, in order to implement a controller of form (4.3) Oustaloup’s band limited frequency domain based rational approximation technique is used in the present paper (Oustaloup et al., 2000).

4.1. Digital realization of fractional order controllers

Fractional control with FO differ-integration can be implemented using the famous Grunwald–Letnikov (G–L) definition, which is nothing but a finite difference approximation of fractional
derivative (integral) with long memory behavior. Though G–L definition is preferred in terms of implementation, the rationale behind choice of Oustaloup based frequency domain rational approximation of FOPID controller is that it can be very easily implemented in real hardware using higher order Infinite Impulse Response (IIR) type digital or analog filters. PID based control schemes are well established in wide range of automation related areas due their simple realization and ease in structure. On the other hand fractional order based control schemes requires rigorous mathematical calculations for realizations and yet they are complex in terms of implementation.

Hence there is threshold between performance levels i.e., in applications where control effort is cheap and where design specifications are not an issue undoubtedly PID controller is preferred. But in cases of accuracy and hard limits over design specifications, FOPID controller will be having good edge rather than PID control scheme. So, appropriate controller for a particular application is left to designer’s choice.

Due to the infinite dimensional nature of fractional order integrator (differentiator) in FOPID controller structure, it creates hardware implementation issues. However, recent research results demonstrated and shown that band-limited implementation of FOPID controllers using higher order rational transfer function approximation of the integro-differential operators gives acceptable performance in industrial applications and automation (Effie, 2011). In this paper, Oustaloup’s recursive approximation that has been implemented to realize fractional integro-differential operators in frequency domain is given by the following equations:

\[
S^o = K \prod_{k=1}^{N} \frac{s + a_k}{s + b_k} + \frac{s}{s + \omega_b}
\]  

(4.5)

where \(N\) represents required number of poles (or zeros) for truncation. \(a_k\) and \(b_k\) represents the zeros and poles of above rational transfer function. Here zeros, poles and gain of the above filter can be recursively evaluated as follows:

\[
a_k = a_b \frac{a_b}{a_h} \left( (k + n + (1/2)(1 - \alpha))/2N + 1 \right) 
\]

(4.6)

\[
b_k = a_b \frac{a_h}{a_b} \left( (k + n + (1/2)(1 + \alpha))/2N + 1 \right) 
\]

(4.7)

\[
K = \frac{a_h^n}{a_b^n} 
\]

(4.8)

In above equations \(\alpha\) is the order of the differ-integration, \((2N + 1)\) is the order of the filter and \((a_b, a_h)\) is the expected fitting range of frequencies, where \(a_b\) is the lower cut-off frequency and \(a_h\) is the upper cut-off frequency. Hence the output of any input signal passing through the filter (4.5) can be termed as approximated fractionally integrated (differentiated) signal with respect to order \(\alpha\). In the current study, 5th \((N=5)\) order Oustaloup’s recursive approximation has been chosen for approximating the integro-differential operators within a frequency band of the constant phase elements (CPEs) as \(\omega \in (10^{-2}, 10^2)\) rad/s.

5. Artificial bee colony (ABC) algorithm

A bee colony can be considered as a swarm intelligence algorithm, whose individual social agents are bees. Bees (precisely hone bees) live in extremely populous colony and maintain a unique elaborate social organization. The exchange of information among bees leads to the formation of tuned collective knowledge. Artificial bee colony (ABC) algorithm is one of well established swarm intelligent based stochastic optimization algorithm, which draws its inspiration from foraging and communication behavior of bees. Since its inception in 2005, ABC has shown remarkable performance in different environments ranging from large scale optimization to complex real world problems. A few challenging applications of ABC algorithm can be found in the areas of clustering (Zhang et al., 2010), economic load dispatch (Nayak et al., 2009), design of digital filters (Karaboga, 2009) and on multi-objective problems as well (Omkar et al., 2011).

In ABC algorithm the foraging artificial bees are categorized into three groups namely employed bees, onlooker bees and scouts. The sum of employed bees and onlooker bees results in total number of bees in a colony (usually they are chosen to be same). The role of employed bee is to hover over a potential food source and to collect, store information regarding a food source in its memory. Onlooker gets information regarding food sources via dances of employed bees. One to one selection is done by onlookers for choosing an employed bee and onlookers try to exploit the food source to gather nectar. For every food source, there is only one employed bee and the employed bee of abandoned food source becomes scout. This scout helps in discovering or exploring sites for more potential food sources. Hence ABC algorithm satisfies the qualities of good optimization algorithm by maintaining a balance between exploitation via employed (onlooker) and exploration via scout bees. In ABC algorithm, each possible solution to the problem is considered as food source and is represented by a \(D\) dimensional real-valued vector, whereas the fitness associated to the source is termed as nectar amount. The only parameters in ABC that are to be set by user are number of bees. Like other population based optimization approaches ABC also progresses towards optimum iteratively.

The algorithm initiates by mapping all employed bees with randomly generated food sources (solutions). The position of \(b_0\) food source (solution) that corresponds to a employed bee in \(D\) dimensional hyperspace can be represented by \(X_i=[x_{i1}, x_{i2}, \ldots, x_{iD}]\). This can be generated between search space bounds with the help of the following equation:

\[
x_{ij} = l_b + rand \times (ubound_j - lbound_j) 
\]

(5.1)

where \(i=1, 2, 3, \ldots, FS; j=1, 2, 3, \ldots, D; FS\) is the number of food sources (equivalent to half to total number of bees) and \(D\) represents total number of variables to be optimized; rand is a random number in the range \((0, 1)\); ubound and lbound correspond to upper and lower bounds of the \(jth\) dimension, respectively. Once employed bee discovers a potential food source it tries to exploit in the vicinity of food source associated to it and updates its search based on the following equation:

\[
x_{new} = x_j + R \times (x_j - x_k) 
\]

(5.2)

Here subscript \(k\) is a randomly chosen index such that \(k \in (1, 2, \ldots, FS)\) and also \(k \neq i\). \(R\) is a uniformly distributed random number in the range \([-1, 1]\). As soon as \(x_{new}\) is produced, a greedy mechanism is applied between \(x_{new}\) and its parent \(x_k\) via fitness value. If the obtained new fitness value is better than the fitness value achieved so far, then the bee moves to this new food source discarding the old source. After successful execution of exploitation phase by employed bees, they share their information regarding food sources with onlooker bees. An onlooker bee selects a particular food source \(X_i\) based on the probability \(P_i\) defined as follows:

\[
P_i = \frac{fit_i}{\sum_{i=1}^{FS} fit_i} 
\]

(5.3)

\(fit_i\) corresponds to fitness value of \(i\)th food source, as the chosen problem is a minimization problem fitness is calculated according to following equation:

\[
fit_i = \frac{1}{1+f(x_i)} 
\]

(5.4)
\( J(X) \) represents objective function to be minimized. Based on above probability relation (analogous to selecting source based on its profitability) onlooker chooses a food source in proximity of previous food source selected by employed bee. There after onlooker tries to exploit a food source making use of Eq. (5.2) and similar greedy mechanism is performed.

The exploitation phase carried out by employed and onlooker bees are performed in round robin fashion. As the search is random in due course of iterative process, it may likely happen that a food source cannot be exploited after some number of iterations and this ultimately leads to delay in optimization process. To eliminate this and to strengthen the search process further, exploration scheme is incorporated in the algorithm via scout bee. Each bee performance is monitored for certain number of cycles (limit), and if the fitness value does not improve then that particular bee role is escalated to scout bee. Food source associated to the scout bee is abandoned and is initialized to random food source via Eq. (5.1). This entire optimization process is continued till the required solution is obtained or termination criteria is obtained.

5.1. Concept of opposition based learning

Literature studies reveal that conventional ABC algorithm suites only for few applications and it requires some additional support to show its exploratory and exploitation power irrespective of problem considered. Inspired by pioneering work of Rahnamayan et al. (2008) we applied opposition based learning (OBL) mechanism in artificial bee colony method to further strengthen its performance. All evolutionary (swarm) based algorithms start with random guesses and computation time towards optimum depends on how much distance these initial solutions are away from optimal solutions. According to the concept of OBL the chance of improving the start of algorithm with a closer solution can be done by checking the opposite solution. To strengthen the algorithm further this concept can be extended to each solution in the current solution.

If \( X = (x_1, x_2, x_3, ..., x_D) \) represents a possible solution or a point in a \( D \)-dimensional hyperspace, where \( x_1, x_2, x_3, ..., x_D \in R \) and \( x_0 \in [\text{bound}_{\text{l}}; \text{bound}_{\text{u}}] \) \( \forall j \in \{1, 2, ..., D\} \). The opposite solution or point associated \( \{j\text{th variable or dimension in a food source associated to a bee}\} \) to \( X \) is given by \( \hat{X} = (\hat{x}_1, \hat{x}_2, \hat{x}_3, ..., \hat{x}_D) \) and is defined as follows:

\[
\hat{x}_i = \text{bound}_l + \text{bound}_u - x_l
\]  

(5.5)

Now with above definition of opposite point the opposition based optimization scheme can be formulated as follows. Assuming \( f(\cdot) \) is fitness function via which candidate fitness is measured and according to the above given definitions of \( X \) and \( \hat{X} \) if \( \hat{f}(\hat{X}) \geq f(X) \) (similar analogy for minimization also) then the point \( X \) can be replaced with \( \hat{X} \); hence the solution and its opposite solution are evaluated simultaneously in order to select fitter one.

5.2. Opposition based artificial bee colony algorithm (OABC)

Opposition based ABC or OABC differs from conventional ABC mainly in two steps. After initializing the employed bees (=FS) to random food sources, opposite population of size FS are also generated using Eq. (5.5) and out of \( 2 \times \text{FS} \) number of solutions best FS solutions are selected via fitness value. Unlike the original Jumping Rate mechanism in Opposition based Differential Evolution (ODE) in this approach we applied opposition concept for continuous search variable in every generation. This is done after completion of onlooker bee phase. As the search is for less number of iterations selecting opposite population by a random probability with respect to Jumping Rate (Rahnamayan et al., 2008) will be worthless. Flow chart of OABC algorithm is provided in Fig. 7.

Pseudo code of OABC algorithm

**User parameters:** Termination Criteria (max iterations), No of Bees, limit.

1. Initialize the food source, which corresponds to initializing the bees to solution, which further means that variables (controller variables) are assigned randomly.
2. Evaluate the opposite food solutions using Eq. (5.5).
3. Evaluate both of the food sources and select best FS food sources, set their roles to employed bees (store the fitness values).
4. Set counter \( \text{Iter} = 1 \)
5. While \( \text{Iter} < \text{Max iterations (or termination criteria)} \) Do
   a. Exploit new food sources via employed bees using Eq. (5.2).
   b. Evaluate the food sources exploited by employed bees via Eq. (5.4) and apply greedy mechanism to select the best source.
   c. Using the probability function defined in Eq. (5.3) obtain probability values.
   d. Assign onlookers for each food source exploited by employed bees.
   e. Exploit new food source using Eq. (5.2) near the food sources selected by onlookers.
   f. Appraise the new food source and apply greedy selection process.
   g. Produce opposite onlooker solutions using Eq. (5.5) and select best FS food sources out of \( 2 \times \text{FS} \) number of onlookers.
   h. Determine abandoned food sources via monitoring them with help of limit parameter and reinitialize them using scout bee mechanism.
   i. Best solutions are memorized.
   j. \( \text{Iter} = \text{Iter} + 1 \)
6. End While
7. Report the best solution obtained so far.

6. Time domain based design of PID/FOPID controllers

In terms of feedback control systems perspective, asymptotic tracking of different types of reference inputs such as ramp and step are one of the prominent control objectives (Tavazoei., 2010). Integral performance indices that are based on system error \( e(t) \) and time \( t \) are considered as reckonable measure of performance of feedback controls. A controller for a system is considered optimum when its parameters are adjusted such that plant meets it design constraints and when index reaches its minimum. The well-known time indices include Integral Absolute Error (IAE), Integral Time Squared Error (ITSE), and Integral Time Absolute Error (ITAE). In this application we considered following time indices for tuning the controllers using GA, PSO, ABC and OABC algorithms. The algorithmic parameters considered for tuning are provided in the Appendix.

\[
J_1 = \text{ITSE} = \int_0^\infty t \cdot e^2(t) \, dt
\]  

(6.1)

\[
J_2 = \text{ITSE} + \text{ISCO} = \int_0^\infty \left[ w_1 \cdot t \cdot e^2(t) + w_2 \cdot u^2(t) \right] \, dt
\]  

(6.2)
As a matter of fact every integral performance has its own advantages in designing control systems. ITSE criterion has an advantage of penalizing error more and due to presence of time multiplication term the oscillations in the systems are damped out quickly. However, ITSE suffers when there is sudden change in set-point leading to large control signal than the ITAE based controllers. On the other hand, the time multiplication term in ITAE term penalizes error more at later stages and helps in reducing settling time ($t_s$) and also the absolute error included will take care of minimization of overshoot ($M_p\%$). The other integrals discussed in Section 2 will also try to optimize performance of controller by minimizing various time domain indices. However, error objective function based methods mere focus is only on error minimization, which ultimately results in very high control signal, which may also result in saturation of actuator leading to integral wind-up. To overcome these problems weighted sum involving integral indices and controller output is preferred. Integral Squared Controller Output (ISCO) is usually preferred for control signal minimization.

For the sake of simplicity and to give equal importance we considered weights $w_1 = w_2 = 1$. A total of four objective functions represented in Eqs (6.1)–(6.4) are used to design optimum PID and FOPID controllers.

In Fig. 8, we provided few surface plots of objective functions for change in integral and derivative orders for stochastically tuned controller gains $K_p$, $K_i$ and $K_d$ using ABC algorithm. These
plots clearly show the complexity an algorithm faces to find controller gains that have to meet the specifications of user. Hence apart from variable gains the variable orders i.e., integral and derivative orders are also to be tuned properly to get required optimal solution. This in turn calls for a robust optimization algorithm, which can provide optimal solution. This problem also enabled us to develop OABC optimization algorithm to obtain optimal solutions within less number of iterations.

6.1. Results and discussion

6.1.1. $J_1$ and $J_2$ based designs

Under constant full load torque condition, (see Appendix for full range of parameters considered) for a reference speed of 55 rad/s and simulation time of $T = 10$ s the speed control of DC motor drive has been performed and analyzed (for illustration purpose in case of ITSE and ITAE Fig. 9(a) and (c) are shown up to time of 2 s). Fig. 9(a) and (b) shows the speed response of DC drive controlled with optimally tuned PID/FOPID controllers, which are designed by minimizing ITSE and ITSE + ISCO objective functions using artificial bee colony algorithms. In case of ITSE or $J_1$ objective function, from Fig. 9(a) and bar chart Fig. 15 it is evident that OABC based FOPID controller provided very less settling time while ABC-PID and OABC-PID controller produced less shoots and less rise time, respectively. Interestingly PSO-PID controller also provided quick rise time but failed in remaining aspects. Although actual goal i.e., minimization of objective function was optimal for OABC-FOPID than ABC, PSO, GA based controllers, it failed to provide optimum set of time indices. In Figs. 15–17 the optimum values are highlighted in bold text.

On the other hand in case of ITSE + ISCO or $J_2$ the results provided by OABC-FOPID are quite promising than the former case. Fig. 9(b) clearly shows that OABC-FOPID provided smooth response, where it provided high overshoots for $J_1$ case. From Figs. 14(c) and 16(d) it is clear that objective value of $J_2$ in case of OABC-FOPID is far less than GA, PSO, ABC based PID and FOPID controllers. Although rise time and settling time of OABC-FOPID are greatly improved while overshoot remained to be poor when compared to ABC tuned FOPID controller (as evident from Fig. 16).

To see the controller performance when sudden load is applied we also recorded the speed responses under full load at $T = 10$ s (simulation time $T = 20$ s) for various controllers. From Fig. 10(a) and (b), OABC based fractional controllers tried to bring back the speed to normal within very less time and also close observation reveals the PID control failed to track the reference speed. This reveals the superiority of FOPID controller in accurate load tracking while maintaining required transient performance.

6.1.2. $J_3$ and $J_4$ based designs

To further improve the performance of proposed controllers, we tried another set of objective functions based on absolute errors i.e., ITAE ($J_3$) and ITAE + ISCO ($J_4$) methods. Fig. 9(c) shows speed response of $J_3$ objective function based DC motor drive under full load torque condition for various controllers. From the plots it is quite obvious that OABC based FOPID controller outperformed the rest of controller’s performances. Peak overshoot (%), settling time (s) of ITAE based FOPID via OABC are far improved when compared to ITSE case. In this case also, in terms of objective function minimization OABC outperformed GA, PSO and ABC methods and can be seen in bar chart depicted in Fig. 17.

Fig. 9(d) delineates speed response of $J_4$ objective function based PID and FOPID controllers for a DC motor drive. In this case OABC tuned FOPID controller outperformed rest of all controllers’ (including ITSE + ISCO) performances in terms of objective function minimization, percentage overshoot, rise time and settling
time. From bar chart Fig. 18(a) it can be seen that OABC-FOPID controller produced no overshoot.

Under sudden loading conditions ITAE (ITAE + ISCO) based OABC-FOPID controllers gave smooth and superior performance. From Fig. 10(c) and (d) it can be seen that they are able to track the reference signal within no time. While PID controllers once again failed to track the reference signal and also they resulted in overshoots followed by slow response. From Figs. 9, 10 and 15–18 it is clear enough that FOPID controllers can show remarkable performance if a proper objective function aided by robust optimization technique is selected. In this application objective functions based on ITAE and ITAE + ISCO are proved to be the most suitable methods when implemented with robust optimization technique like OABC.

6.1.3. Comparison of OABC controller performances over the other competitive designs

In this section we compared and contrasted the OABC based PID/FOPID controller to rest of the controller’s performances.

Case (a): ITSE

Time indices values such as percent overshoot, rise time and settling time of ITSE based controllers are recorded in Fig. 15. Although ABC based PID provided less overshoot it failed to maintain same consistency with respect to rise time and settling and thereby resulting in poor performance (similar results can be seen for GA and PSO based designs also). While on the other hand, apart from maintaining minimum objective function value, OABC based PID controller provided optimum set of time indices. In the case of FOPID also OABC based design outperformed its competitors, but the overshoot remained too high when compared to PID controller.

Case (b): ITSE + ISCO

From Fig. 16 it is clear that every FOPID based design outperformed its corresponding PID controller performance. A close observation also reveals that overshoot of DC motor drive involving FOPID controllers has been drastically reduced than case (a). The inclusion of controller signal minimization helped in restricting the larger shoots thereby resulting in a smooth operation. Fig. 16(d) bar graph clearly shows the effectiveness of OABC in minimizing the objective function both in PID and FOPID. The involvement of fractional operators helped in further minimization of objective function value in the case of FOPID controller.

Case (c): ITAE

In Fig. 17, we reported the time indices and objective values of ITAE based controllers. From Figs. 15(a) and 17(a) it can be seen that overshoot of FOPID controllers in later case decreased. ITAE objective function mostly suited the current application as it provided optimum set of parametric gains, when compared to ITSE. Except in case of overshoot, each and every competitive
algorithm resulted in optimum FOPID controller and OABC based controllers remained to be the best performer. Settling time has been greatly improved and this can be also visualized from speed response plots.

Case (d): ITAE + ISCO

Fig. 18 provides the results of ITAE + ISCO based controller on performances. The inclusion of the additional term to ITAE resulted in zero overshoot in OABC-FOPID controller case. Though rest of time indices are not as good when compared to ITAE, the comparison should be made with respect to ITSE + ISCO not alone with ITAE. All the competitor algorithms i.e., GA, PSO, ABC, OABC based PID/FOPID controller designs produced less rise time, overshoot and also optimal objective function values. As our interest is in load handling also ITAE + ISCO based controller outperformed other controller performances as visualized in Fig. 10.

6.1.4. Impact of PID/FOPID controllers on output control signal

Fig. 11 depicts the control signals produced by ABC and OABC tuned PID/FOPID controllers for various objective functions. From all figures i.e., Fig. 11(a)–(d) it is very clear that FOPID controller tried to provide low intensity control signals, while PID controller very high control signals which are dangerous for practical implementation. A closer observation at Fig. 11 indicates that ABC tuned PID or FOPID controller (for any of objective function from $J_1$–$J_4$) tried to converge to one particular value. Interestingly FOPID controller reached the steady value within less time apart from providing less oscillations and minimum peak value. On other hand OABC tuned FOPID controllers tried to provide control signals that are able to converge to even less value than its counterpart. From Fig. 11(c) and (d) it can be observed that the additional term i.e., ISCO, present in the objective functions $J_2$ and $J_4$ helped the controllers to provide signals that are lower than $J_1$ and $J_2$ based controllers. In this case also OABC-FOPID based controller outperformed rest of all the controllers' performances.

Though the controller design is based on time domain, it is also essential to analyze their frequency domain stability as well. For this purpose, we used the equivalent model depicted in Fig. 5. Bode plots are obtained for OABC tuned PID and FOPID controllers and from Fig. 12, it can be seen that the both the gain and phase margins of FOPID controller are better than normal PID control scheme. In neither of case i.e., for PID controller gave a stable performance. This indicates that FOPID controllers even though failed in few time indices (ITSE case) it maintained good stability margins, which plays a key role in implementation. Also, due to the presence of fractional elements the bandwidth of operation is also increased in case of FOPID controllers. Few applications require robustness to variations in gain, where the speed response should not disturbed when gain is increased. For these kinds of applications designing controllers based on time integrals is complex and hence frequency domain based design is preferred. Section 7 provides deep insights on designing PID and FOPID controller based on frequency-based domain.

Opposition based methods have the ability to provide best values and faster convergence as evident in Fig. 14 and their
7. Frequency domain based design of PID/FOPID controllers

The frequency domain based design of controllers for the DC motor drive is based on the model shown in Fig. 5. Let $P(s)$ be the overall transfer function of the drive considered, then the primary objective is to design a controller $C(s)$, such that the open loop system $G(s)=C(s)P(s)$ will meet the following design criteria.

a. Phase margin criterion

$$\text{Arg}[G(j\omega_c)] = \text{Arg}[C(j\omega_c)P(j\omega_c)] = -\pi + \phi_m \quad (7.1)$$

b. Gain crossover frequency criterion

$$|G(j\omega_c)| = |C(j\omega_c)P(j\omega_c)| = 1 \quad (7.2)$$

c. Robustness to variations in gain criterion

$$\left( \frac{d}{d\omega} \text{Arg}[G(j\omega)] \right)_{\omega = \omega_c} = 0 \quad (7.3)$$

where $\omega_c$ and $\phi_m$ are the gain crossover frequency and phase margin, respectively. Each criterion has its implications on the performance of system. Eq. (7.1) imposes the user defined phase margin ($\phi_m$) at the crossover frequency ($\omega_c$) for the open loop system $G(s)$ to control the most important time index parameter percentage overshoot for the closed loop system. Eq. (7.2) monitors the rise time of system by enforcing the user specified gain crossover frequency. The last and most important criterion refers to Eq. (7.3), it says that derivative of phase of open loop system with respect to frequency at $\omega_c$, is zero i.e., flat or the phase plot is parallel to frequency axis. This criterion is also known as iso-damping property (Das, 2011) and plays very important role in designing high performance control systems. These three design rules can be conglomerated into single objective function and a global optimization method can be applied to obtain the optimum controller parameters:

$$J = (\text{Arg}[C(j\omega_c)] + \text{Arg}[P(j\omega_c)] - 1) + (|C(j\omega_c)P(j\omega_c)| - 1) \times \left( \frac{d}{d\omega} \text{Arg}[C(j\omega)] \right)_{\omega = \omega_c} + \left( \frac{d}{d\omega} \text{Arg}[P(j\omega)] \right)_{\omega = \omega_c} \quad (7.4)$$

Eq. (7.4) represents the sum of phase margin, gain crossover and iso-damping criteria. OABC algorithm is used to minimize the objective function in order to design IO/FO controllers. In order to carry out this optimization task, it is essential to know the frequency responses of plant, controller and also Eq.(7.4) structures.

7.1. Frequency response of DC drive and controllers

7.1.1. DC motor drive model

Based on discussions in Section 2 and also from reduced model shown in Fig. 5, the transfer function of drive considered
The frequency response of the above model (7.5) can be obtained by transforming s domain representation to jω domain.

The constant term \( \frac{R_a K_1}{K_2 K_m T_m} \) is replaced by \( K_1 \):

\[
P(j\omega) = \frac{K_1}{(j\omega) \cdot (1 + j\omega T_1) \cdot (1 + 2j\omega T_2)}
\]  
(7.6)

The corresponding magnitude and phase of DC motor drive model described in (7.6) will be of the following form:

\[
|P(j\omega)| = \frac{K_1}{\sqrt{((2T_2 + T_1) \cdot \omega^2 + (\omega \cdot (1 - 2\omega^2 T_1 T_2))^2}}
\]  
(7.7)

\[
\text{Arg}[P(j\omega)] = \tan^{-1}\left(\frac{1 - 2\omega^2 T_1 T_2}{\omega \cdot (2T_2 + T_1)}\right)
\]  
(7.8)

The derivative of phase of controller with respect to frequency \( \omega \) is given by the following:

\[
\frac{d}{d\omega}\text{Arg}[P(j\omega)] = \frac{-2(T_1 + 2T_2) \cdot (1 + 2\omega^2 T_1 T_2)}{(2\omega T_2 + \omega T_1)^2 + (1 - 2\omega^2 T_1 T_2)^2}
\]  
(7.9)

7.1.2. Integer order PID controller

As said earlier PID controller is a special case of fractional order PID controller, when both integral-derivative orders equals to “1” then Eq. (4.3) reduces to conventional integer order PID controller and its frequency response is given by the following:

\[
C_{PID}(j\omega) = K_p + \frac{K_i}{(j\omega)} + K_d(j\omega) = K_p + j\left(\frac{-K_i + K_d \omega^2}{\omega}\right)
\]  
(7.10)

The magnitude and phase of PID controller is described in Eq. (7.10) can be calculated as follows:

\[
|C_{PID}(j\omega)| = \sqrt{K_p^2 + \left(\frac{-K_i + K_d \omega^2}{\omega}\right)^2}
\]  
(7.11)
The derivative of phase of controller (7.10) with respect to frequency $\omega$ is as follows:

$$\frac{d}{d\omega}(\text{Arg}[C_{PID}(j\omega)]) = \frac{K_p(K_i + K_o\omega^2)}{(K_p)^2 + (-K_i + K_o\omega^2)^2}.$$  

(7.13)

### 7.1.3. Fractional order PID controller

The frequency response of Fractional order PID controller defined in Eq. (4.3) is given by the following set of equations:

$$C_{FOPID}(j\omega) = K_p + K_i(j\omega)^{-\lambda} + K(D(j\omega)^\mu)$$

$$= K_p + K_i\omega^{-\lambda}\left(\cos\frac{\lambda\pi}{2} - j\sin\frac{\lambda\pi}{2}\right) + K(D\omega^\mu)\left(\cos\frac{\mu\pi}{2} + j\sin\frac{\mu\pi}{2}\right)$$

$$= \left(K_p + K_i\omega^{-\lambda}\cos\frac{\lambda\pi}{2} + K(D\omega^\mu)\cos\frac{\mu\pi}{2}\right)$$

$$+ j\left(K(D\omega^\mu)\sin\frac{\mu\pi}{2} - K_i\omega^{-\lambda}\sin\frac{\lambda\pi}{2}\right)$$

(7.14)

The magnitude, phase, and derivative of phase with respect to $\omega$ of FOPID controller are shown in the following equations:

$$|C_{FOPID}(j\omega)| = \sqrt{(K_p + K_i\omega^{-\lambda}\cos\frac{\lambda\pi}{2} + K(D\omega^\mu)\cos\frac{\mu\pi}{2})^2}$$

(7.15)

$$\text{Arg}[C_{FOPID}(j\omega)] = \tan^{-1}\left(\frac{K(D\omega^\mu)\sin(\mu/2) - K_i\omega^{-\lambda}\sin(\lambda/2)}{K_p + K_i\omega^{-\lambda}\cos(\lambda/2) + K(D\omega^\mu)\cos(\mu/2)}\right)$$

(7.16)

$$\frac{d}{d\omega}(\text{Arg}[C_{FOPID}(j\omega)]) = \frac{(K(D\omega^\mu)\sin(\mu/2) - K_i\omega^{-\lambda}\sin(\lambda/2)) + (K_pK(D\omega^\mu)^{-\lambda-1}\sin((\lambda+\mu)/2))}{(K_p + K_i\omega^{-\lambda}\cos(\lambda/2) + K(D\omega^\mu)\cos(\mu/2))^2}$$

(7.17)

### 7.2. Impact of frequency domain based controllers on performance of DC motor drive

To obtain optimum controller parameters, the objective function (7.4) is defined using frequency response of drive; controller structures Eqs. (7.5)–(7.17) and is minimized with the help of OABC algorithm. Based on the bode plots obtained in Fig. 12 the frequency design criteria are considered to be crossover frequency $\omega_c = 4.5\text{ rad/s}$ and phase margin of $\varphi_m = 80^\circ$. The controller variables search space and the algorithm parameters are provided in Appendix B. To verify the robustness of algorithm via statistical norms, mean and standard deviation (std) values of evolutionary algorithms used are recorded for 25 independent trails for both the controller structures.

#### 7.2.1. Integer order PID control

For the design criteria mentioned the resulting integer PID controller is given in Eq. (7.18). It is evident from bode plots [Fig. 19] that PID controller tried to meet the design criteria while it was not successful in maintaining a complete flat phase around gain crossover frequency. Robustness against gain variation is observed by increasing motor gain in steps and resulting step responses are depicted in Fig. 20. Though PID controller tried to with stand variations in gain by providing a response closer to normal one, for large gain variation it failed to track the reference:

$$C_{PID}(s) = 2.1081 + \frac{0.021}{s} + 0.073s.$$  

(7.18)
Fig. 14. Convergence of ABC and OABC algorithms towards optimum in case of (a) ITSE, (b) ITSE + ISCO, (c) ITAE, and (d) ITAE + ISCO based PID/FOPID controllers for DC motor drive.

Fig. 15. Bar chart showing time indices parameters and objective functions for GA, PSO, ABC and OABC methods for ITSE ($J_1$) tuned PID and FOPID controllers.
Fig. 16. Bar chart showing time indices parameters and objective functions for GA, PSO, ABC and OABC methods for ITSE + ISCO ($J_2$) tuned PID and FOPID controllers.

Fig. 17. Bar chart showing time indices parameters and objective functions for GA, PSO, ABC and OABC methods for ITAE ($J_3$) tuned PID and FOPID controllers.
Table 3 shows various time indices parameters for increase in gain. As the gain increased the time indices parameter values decreased. Due to presence of iso-damping nature overshoot hasn’t increased though the gain is increased to 80%.

7.2.2. Fractional order PID control

It can be observed from bode plot Fig. 21 that fractional order PID controller perfectly met the design criteria while maintaining flat phase around gain crossover frequency. Unlike to the PID controller, FOPID maintained good robustness against gain variation.

Further from Table 3 it is evident that the FOPID outperformed PID controller in terms of percentage overshoot, rise time and settling time and (also objective function value). The optimal frequency domain based FOPID controller is given in Eq. (7.19). From Figs. 21 and 22 it can be seen that Fractional elements helped to provide iso-damping characteristics for wide range of gain.

Further from Table 3 it is evident that the FOPID outperformed PID controller in terms of percentage overshoot, rise time and settling time and (also objective function value). The optimal frequency domain based FOPID controller is given in Eq. (7.19). From Figs. 21 and 22 it can be seen that Fractional elements helped to provide iso-damping characteristics for wide range of gain.

$$C_{FOPID}(s) = 2.0563 + \frac{0.0223}{s^{0.007}} + 0.0219s^{0.0102}$$  (7.19)

The fractional elements apart from maintaining wide range of iso-damping characteristics also tried to maintain 0% percent
overshoot for increase in gain. In this frequency based design also FOPID controller outperformed PID controller in all time domain indices. Corresponding objective function values of PID and FOPID tuned via OABC method is provided in Table 3.

<table>
<thead>
<tr>
<th>Controller</th>
<th>J3 — ITAE</th>
<th>J4 — ITAE + ISCO</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>KP</td>
<td>KI</td>
</tr>
<tr>
<td>GA-PID</td>
<td>4.733</td>
<td>3.8575</td>
</tr>
<tr>
<td>GA-FOPID</td>
<td>5.7938</td>
<td>9.92</td>
</tr>
<tr>
<td>PSO-PID</td>
<td>4.3218</td>
<td>3.5545</td>
</tr>
<tr>
<td>PSO-FOPID</td>
<td>6.8286</td>
<td>9.9691</td>
</tr>
<tr>
<td>ABC-PID</td>
<td>5.011</td>
<td>3.5528</td>
</tr>
<tr>
<td>ABC-FOPID</td>
<td>6.9671</td>
<td>10</td>
</tr>
<tr>
<td>OABC-PID</td>
<td>8.8481</td>
<td>5.3892</td>
</tr>
<tr>
<td>OABC-FOPID</td>
<td>9.8759</td>
<td>6.1824</td>
</tr>
</tbody>
</table>

8. Conclusions

In this research, the performance of opposition based artificial bee colony algorithm (OABC) has been investigated for designing PID and FOPID controllers for speed control of chopper fed DC motor drive. Objective functions based on ITSE and ITAE and ISCO are considered for designing the controller in time domain and an objective function based on frequency rules is considered for designing controller in frequency domain. Extensive analysis over the results obtained in terms of performance and stability revealed that for the application considered ITAE methods proved to be worth full than ITSE creation. This paper also addressed the added advantages of time domain methods and also frequency domain method of design (to obtain iso-damping characteristics) and the ultimate decision is left to the user. Our studies also revealed that
fractional controllers when designed optimally could outperform conventional PID controllers.

Our future research would focus on designing multi-objective framework for controller tuning to achieve still promising results under different circumstances like levying constraints on the control signals, time domain based fractional order objective functions, etc.

Acknowledgments

This work was supported in the framework of the IT4 Innovations Center of Excellence project, Reg. no. CZ.1.05/1.1.00/02.0070 by operational program Research and Development for Innovations funded by the Structural Funds of the European Union and state budget of the Czech Republic, EU.

The authors are extremely thankful to Scientist Shantanu Das (Reactor Control Division at Bhabha Atomic Research Center, India) (Das, 2011) for his tremendously valuable comments on the improvement of this paper and also his discussions on concepts of iso-damping properties.

Appendix

A. DC motor drive specifications (Gopakumar)
Power = 300 kW
Base speed = 500 rpm; 55 rad/s
Max rated voltage ($V_a$) = 460 V
Max rated current ($I_a$) = 690 A
Current limit = 1200 A
Inductance ($L_a$) = 0.7026 mH
Moment of inertia ($J$) = 84 kg m²

<table>
<thead>
<tr>
<th>DC gain increase (%)</th>
<th>Rise time (PID)</th>
<th>Rise time (FOPID)</th>
<th>Overshoot (% (PID))</th>
<th>Overshoot (%) (FOPID)</th>
<th>Settling time (PID)</th>
<th>Settling time (FOPID)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>0.6110</td>
<td>0.4072</td>
<td>0.8157</td>
<td>0</td>
<td>1.0994</td>
<td>0.7473</td>
</tr>
<tr>
<td>20% increase</td>
<td>0.5265</td>
<td>0.3273</td>
<td>0.7335</td>
<td>0</td>
<td>0.9016</td>
<td>0.6013</td>
</tr>
<tr>
<td>40% increase</td>
<td>0.4639</td>
<td>0.2703</td>
<td>0.6374</td>
<td>0</td>
<td>0.8938</td>
<td>0.4954</td>
</tr>
<tr>
<td>60% increase</td>
<td>0.4151</td>
<td>0.2278</td>
<td>0.4598s</td>
<td>0</td>
<td>0.8260</td>
<td>0.4140</td>
</tr>
<tr>
<td>80% increase</td>
<td>0.3758</td>
<td>0.1953</td>
<td>0.3171</td>
<td>0</td>
<td>0.7715</td>
<td>0.3490</td>
</tr>
</tbody>
</table>

$J_{PID} = 3.0171 \times 10^{-3}$ (5.22581e$^{-04}$)

$J_{FOPID} = 2.4818 \times 10^{-2}$ (9.05335e$^{-03}$)

Fig. 21. Bode plot of OABC tuned FOPID controller for DC motor drive.

Fig. 22. Step response of FOPID controlled DC motor drive up to 80% increase in gain.
Back EMF constant \( (K_{em}) = 8.5 \text{ V s/rad} \)
Armature resistance \( (R_a) = 0.02342 \Omega \)
Current feedback \( (T_f) = 3.5 \text{ ms} \)
Tachometer constant \( (T_p) = 25 \text{ ms} \)
Electromechanical time constant \( (T_m) = 2.58 \times (K_{em})^2 = 27.55 \text{ ms} \)
Controller time constant \( (T_{c}) = L_p/R_a = 30 \text{ ms} \)
Armature time constant \( (T_a) = T_p = 30 \text{ ms} \)
Chopper constant \( (K_c) = \max \text{ rated voltage} / 10 = 46 \)
Speed loop filter gain \( (K_f) = 10/55 = 0.19 \)
Current loop filter gain \( (K_{cl}) = 10/\text{current limit} = 0.0083 \)
Controller gain \( (K_c) = T_a \times R_a / (2 \times K_c \times T_p) = 0.2618 \)

### B. Algorithmic parameters

#### Binary GA (Biswas et al., 2009)
Initial population size = 50
No. of bits per gene = 50
Mutation probability = 0.01
Crossover probability = 0.6

#### PSO (Zamani et al., 2005)
Population size = 30

#### ABC and OABC
No. of bees \( (N_b) = 20 \)
No. of food sources \( (S_F) = N_b/2 \)
Limit = 30

#### Boundary settings
No. of iterations = 100
Run time = 25 independent trails
Controller gains range = \( 0 \leq K_P, K_I, K_D \leq 10 \)
Fractional orders range = \( 0 \leq \lambda, \mu \leq 1 \)

### C. Computer settings
All the simulations have been carried out by developing dedicated software programmes in MATLAB 7.9 version, compiled by Intel core i3 processor, and ran on 2.40 GHz computer with 2 GB of memory under windows 7 environment.

### References


### Notes


### B. Algorithmic parameters

#### Binary GA (Biswas et al., 2009)
Initial population size = 50
No. of bits per gene = 50
Mutation probability = 0.01
Crossover probability = 0.6

#### PSO (Zamani et al., 2005)
Population size = 30

#### ABC and OABC
No. of bees \( (N_b) = 20 \)
No. of food sources \( (S_F) = N_b/2 \)
Limit = 30

#### Boundary settings
No. of iterations = 100
Run time = 25 independent trails
Controller gains range = \( 0 \leq K_P, K_I, K_D \leq 10 \)
Fractional orders range = \( 0 \leq \lambda, \mu \leq 1 \)

### C. Computer settings
All the simulations have been carried out by developing dedicated software programmes in MATLAB 7.9 version, compiled by Intel core i3 processor, and ran on 2.40 GHz computer with 2 GB of memory under windows 7 environment.

### References


### Notes