Int. J. Mach. Learn. & Cyber.
DOI 10.1007/s13042-012-0136-2

ORIGINAL ARTICLE

A Hybrid Differential Artificial Bee Colony Algorithm based tuning of fractional order controller for Permanent Magnet Synchronous Motor drive

Anguluri Rajasekhar · Ajith Abraham · Millie Pant

Received: 23 June 2012 / Accepted: 8 November 2012

Abstract In this paper a novel Hybrid Differential Artificial Bee Colony Algorithm (HDABCA) has been proposed for designing a fractional order proportional-integral (FO-PI) speed controller in a Permanent Magnet Synchronous Motor (PMSM) drive. FO-PI controllers' parameters involve proportionality constant, integral constant and integral order, and hence its design is more complex than that of the usual Integral-order proportional-integral controller. To overcome this complexity in designing, we had used the proposed hybrid algorithm, such that all the design specifications of the motor are satisfied. In order to digitally realize the FO-PI controller, an Oustaloup approximation method has been used. Simulations and comparisons of proposed HDABCA with conventional methods and also other state-of-art methods demonstrate the competence of the proposed approach, especially for actuating fractional order controller for integer order plants.

Keywords PMSM · FO-PI speed controller · Differential evolution · Artificial Bee Colony Algorithm

1 Introduction

Recent studies had shown that with the advancement of control theories, power electronics and microelectronics in connection with new motor design and magnetic materials since 1980s, electrical (AC) drives are making tremendous impact in the area of various high-performance variable speed control systems [1, 2]. Among AC drives PMSM with high energy permanent magnetic materials like "Neodymium Iron Boron" (Nd–Fe–B) or "Samarium-cobalt alloy" (Sm2Co17) provide fast dynamics with the applications if controlled properly.

Electrically excited field windings are being replaced by Permanent Magnets because of their advantages like elimination of brushes, slip-rings, rotor copper losses which yields high efficiency and produces constant flux. The design criteria of synchronous servo motors, to be used in industrial applications differ from that of conventional synchronous motors because of many practical constraints of which few of them are high air-gap flux density, high power to weight ratio, low noise followed by low inertia, small torque ripple and compact design in wide range of speed [2]. These requirements can be met well by PMSM with sinusoidal flux distribution due to lower torque ripples.

Now-a-days, the vector control techniques has made it possible to apply the PMSM Drives in high-performance

Part of this work was published in the Proceedings of the Third World Congress on Nature and Biologically Inspired Computing (NABIC-2011) [27].

A. Rajasekhar
Department of Electrical and Electronics Engineering, National Institute of Technology, Warangal 506004, Andhra Pradesh, India
e-mail: rajasekhar.anguluri@ieee.org

A. Abraham (✉)
Faculty of Computer Science and Electrical Engineering, VSB—Technical University of Ostrava, Ostrava, Czech Republic
e-mail: ajith.abraham@ieee.org; abraham.ajith@gmail.com

A. Abraham
Machine Intelligence Research Labs (MIR Labs), Auburn, WA, USA

M. Pant
Department of Paper and Pulp Technology, Indian Institute of Technology, Roorkee 247667, India
e-mail: millifpt@iitr.ernet.in

Published online: 29 November 2012

industrial applications where only D.C motor drives were previously available, to achieve fast four-quadrant operation [3]. In order to get an optimum motor performance, a very effective control system is needed. Although many possible solutions like nonlinear, adaptive control are available [4], the market of electrical drives doesn't justify the expense needed to implement such sophisticated solution in industrial drives and Proportional-Integral (PI) based control system scheme still remains the most widely adopted solution. Such a propensity is supported by a fact that, though simple, a PI control achieves high performance when optimally designed [5]. PI controllers have been widely used for decades in industries for process control applications and their reason for their wide popularity lies in the simplicity of design and performances including low percentage overshoot and low maintenance cost [5].

An elegant way of enhancing the performance of PI controller is to use fractional-order controllers. Dynamic systems based on fractional order calculus have been a subject of extensive research in recent years, since the concept of fractional-order controllers was proposed by Podlubny [6, 21], who demonstrated their effectiveness in actuating desired fractional order system responses [6]. Fractional Order Proportional Integral (FO-PI) controller is a convenient fractional order structure that has been employed for control purposes. In an FO-PI controller (in general PI^{λ}) I-operations are usually of fractional order; therefore besides setting K_P, K_I we have another parameter i.e., order of fractional integration λ. If $\lambda = 1$ it is an integral order PI (IO-PI) controller. Hence, fractional order controller is a generalized version of IO-PI controller. Finding appropriate set of values for these three parameters to achieve optimum performance of PMSM drive in three dimensional hyper-space calls for real parameter optimization. In this approach our tuning method focuses on minimizing the objective function i.e., Integral Time Absolute Criterion (ITAE) criterion.

The optimization model developed involves minimizing the error which is the difference of reference speed and obtained speed. Mathematically, the objective function has a multimodal error surface which limits the application of conventional gradient based methods. Consequently attention is given to global optimization techniques like Genetic Algorithm (GA) [7], Bacterial Foraging Optimization [8], Particle Swarm Optimization [9, 10]. In the present study we have proposed a hybrid of Differential Evolution (DE) [11] and Artificial Bee Colony Algorithm (ABC) [12] to deal with this optimization model.

DE and ABC are a part of nature inspired search techniques used for solving complex and intricate optimization problems that are otherwise difficult to solve by traditional techniques. These algorithms have been successfully applied to a wide range of engineering design problems [13–15] etc.,

However, despite having several favorable features, it has been observed that these techniques sometimes do not perform up to the mark when used in their basic form. Several studies can be found in literature [13, 15] devoted to improve the performance of DE and ABC. In the present study we have tried to combine the two, so as to develop a hybrid algorithm which not only has a fast convergence but also maintains the quality of solution. The proposed algorithm is named Hybrid Differential Artificial Bee Colony Algorithm (HDABCA). HDABCA has been first tested on a set of benchmark problems and based on the significant performance of HDABCA we had used this method in designing an optimum IO-PI and FO-PI controllers for PMSM drive.

The rest of paper is organized as follows. Section 2 introduces the basics of PMSM motor followed by a mathematical modeling of the motor. Section 3 presents the problem formulation and brief introduction to fractional calculus followed by approximation method. Section 4 outline the proposed HDABCA method. Sections 5 and 6 illustrates the simulations and comparisons of proposed approach for designing FO-PI as well as IO-PI controllers. Finally, the conclusions drawn from the present study are provided in Sect. 7.

2 Permanent Magnet Synchronous Motor

In general, PMSM with approximately sinusoidal back electromotive force (i.e., back EMF) can be broadly categorized into two types (1) Interior (or buried) Permanent Magnet Synchronous Motors (IPMSM) (2) Surface mounted Permanent Magnet Synchronous Motors (SPMSM). In this paper we have considered the SPMSM. The cross-sectional layout of SPMSM is shown in Fig. 1.

In this type of motor the magnets are mounted on the surface of the motor. Because the incremental permeability of these magnets is between 1.02 and 1.20 relative to external fields, the magnets have high reluctance and accordingly the SPMSM can be considered to have large

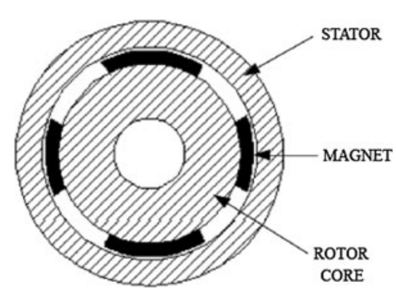

Fig. 1 Structure of Permanent Magnet Synchronous Motor [12]

and effective uniform air-gap. This property makes the saliency effect negligible. Thus quadrature axis synchronous inductance of SPMSM is equal to its direct axis inductance. As a result magnetic torque can only be produced by SPMSM, which arises from the interaction of magnet flux and quadrature axis current. The stator carries a three-phase winding which produces a near sinusoidal distribution of magneto motive force based on the value of stator current. They have the same role as the field winding in a synchronous machine except that their magnetic field is constant and there is no control on it [16].

2.1 Dynamic *d-q* axis of PMSM

By considering all the parameters, equivalent circuit of PMSM can be represented in direct-quadrature (*d-q* axis) reference frame. In this paper we have considered the assumptions of [17]. By taking these assumptions the modified dynamic equivalent circuit of SPMSM is obtained and shown in Figs. 2 and 3.

2.2 Mathematical model of PMSM drive

The PMSM drive model consists of a Pulse Width Modulation (PWM) inverter, a PWM generator, a current controller followed by speed controller and is also embedded with speed/position estimator. The schematic representations of these components are shown in Fig. 4. The PMSM drive receives power from three-phase AC supply and runs mechanical load at desired speed. The developed model of the drive system is used for design in current and speed controllers. The mathematical model of PMSM in *d-q* synchronously rotating frame of reference can be obtained from synchronous machine model. The PMSM can be represented by the set of following nonlinear [17] differential equations.

$$v_{sd} = r_s i_{sd} + p\lambda_{sd} - \omega_e \lambda_{sq} \tag{1}$$

$$v_{sq} = r_s i_{sq} + p\lambda_{sq} + \omega_e \lambda_{sd} \tag{2}$$

$$\lambda_{sd} = L_d i_{sd} + \lambda_m \tag{3}$$

$$\lambda_{sq} = L_q i_{sq} \tag{4}$$

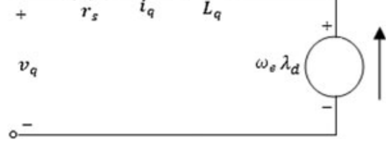

Fig. 2 Dynamic *q* axis equivalent circuit of PMSM

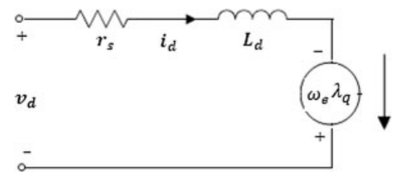

Fig. 3 Dynamic *d* axis equivalent circuit of PMSM

$$T_e = \frac{3}{2}\frac{P}{2}[\lambda_m i_{sq} + (L_d - L_q) i_{sd} i_{sq}] \tag{5}$$

$$T_e = J\frac{2}{p}\frac{d\omega_e}{dt} + B\frac{2}{P}\omega_e + T_l \tag{6}$$

$$\omega_e = P\omega_r/2 \tag{7}$$

$$p\theta_r = \frac{2}{P}\omega_e \tag{8}$$

where v_{sq}, v_{sd}, i_{sq}, i_{sd} are *d-q* axis voltages and currents respectively. L_d, L_q are *d-q* axis inductances; λ_{sq}, λ_{sd} are *d-q* axis flux linkages; and ω_e, r_s are electrical speed of motor, and stator resistance respectively. λ_m is the constant flux linkage due to rotor permanent magnet; T_e is the electromagnetic torque; T_l is the load torque; P represents number of poles; p is the differential operator; B is the damping coefficient; θ_r is the rotor position; ω_r is the rotor speed; and J is the moment of inertia.

For constant flux operation when $i_{sd(ref)}$ equals zero [13], in vector-control technique or Field Oriented Control (FOC) the equations of PMSM are modified as

$$pi_{sd} = (v_{sd} - r_s i_{sd} + \omega_e L_q i_{sq})/L_d \tag{9}$$

$$pi_{sq} = (v_{sq} - r_s i_{sq} - \omega_e L_d i_{sd} - \omega_e \lambda_m)/L_q \tag{10}$$

$$\frac{d\omega_e}{dt} = \frac{1}{J}\left[\frac{P}{2}(T_e - T_l) - B\omega_e\right] \tag{11}$$

$$T_e = \frac{3}{2}\frac{P}{2}[\lambda_m i_{sq}] = K_t i_{sq} \tag{12}$$

The vector control method i.e., $i_{sd} = 0$ is widely used in the surface-mounted Permanent Magnet Synchronous Motor or salient motor which does not need a weak magnetic control. The method that $i_{sd} = 0$ has some

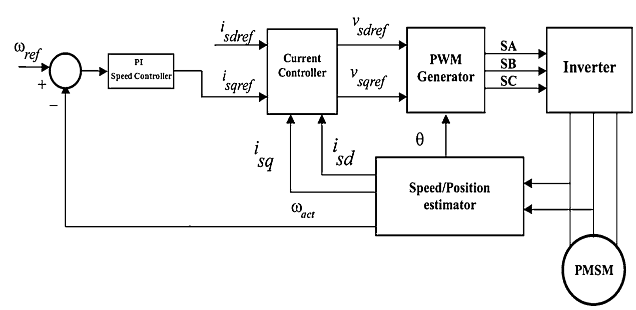

Fig. 4 *Block diagram* of PMSM Drive

advantages such as making simple control structure and the torque is directly proportional to the current of q axis.

3 Problem formulation

Fractional Calculus (FC) is the branch of Mathematics, having 300 years of history. Recently this theory was applied to many fields of science and engineering [18, 20]. FC is a generalization of ordinary differential calculus which considers the possibility of taking real number power of differential and integration operator [19]. There are many ways to describe fractional-order integrals and derivatives. The main concept of FC lies in developing a functioning operator D which is known as differ-integrator operator, according to Riemann-Liouville definition [22] it is mathematically defined as

$$ {}_aD_t^{\alpha} f(t) = \frac{1}{\Gamma(m-\alpha)} \left(\frac{d}{dt}\right)^m \int_a^t \frac{f(\tau)}{(t-\tau)^{1-(m-\alpha)}} d\tau \tag{13} $$

where $\Gamma(\cdot)$ is the Euler's γ function in the range of $m - 1 < \alpha < m$. Another alternative method of defining this D is by using the concept of fractional differentiation by Grünwald-Letnikov, which is given by

$$ {}_aD_t^{\alpha} f(t) = \lim_{h \to 0} \frac{1}{\Gamma(\alpha) h^{\alpha}} \sum_{k=0}^{(t-a)/h} \frac{\Gamma(\alpha+k)}{\Gamma(k+1)} f(t-kh). \tag{14} $$

Laplace transform is the additional tool used in this fractional calculus. The Laplace transform of nth ($n \in R_+$) derivative of a signal $x(t)$ in relaxed mode i.e., $t = 0$ is given as: $L\{D^n x(t)\} = s^n X(s)$. Here s is Laplacian operator. So, a fractional order differential equation for given input and output signals $u(t)$ and $y(t)$ (relaxed at $t = 0$) is expressed as a transfer function of form

$$ G(s) = \frac{a_1 s^{\alpha_1} + a_2 s^{\alpha_2} + \cdots + a_{m_A} s^{\alpha_{m_A}}}{b_1 s^{\beta_1} + b_2 s^{\beta_2} + \cdots + b_{m_B} s^{\alpha_{m_B}}} \tag{15} $$

where $(a_m, b_m) \in R^2, (\alpha_m, \beta_m) \in R_+^2, \forall (m \in N)$. The generalized form of fractional order PI controller is the PI^{λ} controller, which involves an fractional integrator of order λ (can be any real number). The controller signal $u(t)$ can then expressed in time domain as

$$ u(t) = K_P e(t) + K_I D^{-\lambda} e(t) \tag{16} $$

In the frequency domain the transfer function of this controller is given as

$$ G_c(s) = K_P + \frac{K_I}{s^{\lambda}} \tag{17} $$

The interpretation of s^{λ} is that, on a semi-log plane, there is a line having slope of $-20\ \lambda$ dB/dec.

Where $\lambda = +1$ implies normal PI controller and $\lambda = 0$ implies proportional gain. All these different forms of PI controller are the special cases of fractional PI^{λ} controller. It is very interesting to note that FO-PI controller generalizes the IO-PI controller and expands it from point to plane. This will add more flexibility to controller design and the controlling of real word process is more precise and accurate [19].

It is very important to implement band-limit for fractional order controller in real applications, and also finite dimensional approximation of the fractional controller should be in a appropriate range of frequencies. The following section will present the brief outline of Oustaloup method [20] for realization of the controller.

3.1 Digital realization of FOPI controller

The usual way of using transfer functions including fractional orders of "s" is to approximate it to integer order transfer functions. To actually approximate a fractional transfer function, an integer transfer function would have to involve an infinite number of poles and zeros (which is very difficult). But it is possible to obtain logical approximations with a finite number of zeros and poles.

Out of many real approximation methods, one of the well-known approximations is by Oustaloup method which uses recursive distribution of poles and zeros. According to Oustaloup the transfer function is given by

$$ s^{\alpha} \approx k \prod_{n=1}^{N} \frac{1 + (s/\omega_{z,n})}{1 + (s/\omega_{p,n})}. \tag{18} $$

The approximation is valid in the frequency range of $[\omega_l, \omega_h]$. The value of poles and zeros (N) is fixed, The required performance of the approximation mainly depends on: truncated values causing simpler approximations, but may lead to ripples in both phase and gain regions. These ripples may be functionally removed by increasing poles and zeros count, ultimately leading to heavier computation. Required frequencies of poles and zeros in Eq. (18) is given according to

$$ \omega_{z,1} = \omega_l \sqrt{\eta} \tag{19} $$

$$ \omega_{p,n} = \omega_{z,n} \epsilon, \quad n = 1, \ldots, N \tag{20} $$

$$ \omega_{z,n+1} = \omega_{p,n} \eta, \quad n = 1, \ldots, N-1 \tag{21} $$

$$ \epsilon = (\omega_h/\omega_l)^{v/N} \tag{22} $$

$$ \eta = (\omega_h/\omega_l)^{(1-v)/N}. \tag{23} $$

In the case of $\alpha < 0$, this can be handled by reversing Eq. (18).

3.2 Formulation of the objective function

Various performance criteria like Integral Absolute Error (*IAE*), Integral Squared Error (*ISE*) etc., are available in literature for designing the controllers. The main drawback of using *ISE* and *IAE* criteria is a dynamic response with a relatively less overshoot but a heavy settling time because they will calculate the errors uniformly over time [23]. Integral-Time-Weighted-Squared-Error (*ITSE*) on other hand does not have these draw backs, but it cannot ensure to have desirable stability (and computationally complex). Therefore, we employed Integral-time-weighted-absolute-error (*ITAE*) which has an advantage of producing lesser oscillations and overshoot along with less settling time. Mathematically defined as

$$ITAE = \int_0^t t(|\omega_{ref} - \omega_{act}|) \cdot dt = \int_0^t t(|e(t)|) \cdot dt. \qquad (24)$$

The performance of the drive depends on the fractional controller parameters values which indeed depend on the objective function to be minimized. So, to get a optimal set of parametric values for K_P, K_I, and λ to meet the user defined specifications for a given process call for real parameter optimization in three-dimensional hyperspace.

4 Hybrid Differential Artificial Bee Colony Algorithm

In this section before describing the proposed HDABCA algorithm, we give a brief overview of ABCA and DE algorithms.

4.1 Artificial Bee Colony

Artificial Bee Colony Algorithm (ABCA) is a swarm intelligence search technique inspired by the foraging behavior of honey bees. It was proposed by Karaboga [12] for solving multivariable and multi-modal continuous functions. The ABCA classifies the foraging artificial bees into three groups namely *employed bees, onlooker bees* and *scouts*. The first half colony consists of the *employed bees* and second half consists of *onlooker bees*. A bee that is currently searching for food or exploiting a food source is called an *employed bee* and a bee waiting in the hive for making decision to choose a food source is called an *onlooker*. For every food source, there is only one employed bee. The employed bee of abandoned food source becomes scout. In ABCA, each solution to the problem is considered as *food source* and represented by a *D*-dimensional real-valued vector, where the fitness of the solution corresponds to the *nectar amount* of associated food resource. Like other swarm based algorithms, ABCA is also an iterative process. It starts with population of randomly distributed solutions or food sources.

The algorithm starts by initializing all employed bees with randomly generated food sources. In general the position of *i*th food source that corresponds to the solutions in the search space are represented as $X_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$, and is generated by following equation.

$$x_{ij} = lb_j + rand * (ub_j - lb_j)$$

where $i = 1, 2, \ldots, FS, j = 1, 2, \ldots, D$. *FS* is the number of food sources and *D* is the number optimization parameters. Where *rand* is a random number in range of [0 1]; ub_j and lb_j are upper and lower bounds for the *j*th dimension respectively. After the information is shared by the employed bees, *onlooker* bees go to the region of food source at X_i based on the probability P_i defined as

$$P_i = \frac{fit_i}{\sum_{k=1}^{FS} fit_k}. \qquad (25)$$

FS is total number of food sources. Fitness value fit_i is calculated by using following equation.

$$fit_i = \frac{1}{1 + f(X_i)} \qquad (26)$$

here $f(X_i)$ is the objective function to be minimized, in our problem ITAE. The *onlooker* finds its *food source* in the region X_i, by making use of following equation

$$x_{new} = x_{ij} + r * (x_{ij} - x_{kj}) \qquad (27)$$

where $k \in (1, 2, \ldots, FS)$ such that $k \notin i$ and $j \in (1, 2, \ldots, D)$ are randomly chosen indexes. *r* is a uniformly distributed random number in the range [−1, 1].

If the obtained new fitness value is better than the fitness value achieved so far, than the bee moves to this new food source leaving this old one otherwise it retains the old food source. When all employed bees have completed this process, the information is shared with onlookers. Each of the onlookers selects a food source according to the probability given above. By this scheme good sources are well accommodate with onlookers. Each bee will search for a better food source for a certain number of cycles *(limit)*, and if the fitness value doesn't improve then that particular bee becomes a Scout bee. A food source is initialized to that scout bee randomly and the search process continues.

4.2 Differential evolution

Differential Evolution was proposed by Price and Storn [24]. It is also a population based algorithm like genetic algorithms (GA), using the similar operators: crossover, mutation and selection. The main difference in constructing better solutions is that GA rely on crossover while Differential Evolution (DE) relies on mutation operator.

The search process of DE is as follows: At first, all the individuals are initialized with uniformly distributed random numbers and evaluated using the objective function provided. Then the following sequential steps are executed until it reaches optimum solution or pre defined termination criterion is found.

The working of DE is as follows, In a D-dimensional search space, for each target vector $x_{i,g}$ a mutant vector is generated by

$$v_{i,g+1} = x_{r_1,g} + F * \left(x_{r_2,g} - x_{r_3,g}\right) \tag{28}$$

where $r1, r2, r3 \in \{1, 2, \ldots, NP\}$ are randomly chosen integers from population of size emphNP, different from each other and also different from the current running index i. The parameter $F(>0)$ is known as scaling factor which controls the amplification of the differential evolution $\left(x_{r_2,g} - x_{r_3,g}\right)$. In order to increase the diversity of the perturbed parameter vectors, crossover is introduced. The parent vector is mixed with the mutant vector to produce a trail vector $u_{ji,g+1}$, given by

$$u_{ji,g+1} = \begin{cases} v_{ji,g+1}, & \text{if } \left(rand_j \leq CR\right) \text{or} (j = j_{rand}) \\ x_{ji,g}, & \text{if } \left(rand_j > CR\right) \text{and } (j \neq j_{rand}) \end{cases} \tag{29}$$

Here $j = 1, 2, \ldots, D; rand_j \in [0, 1]$; CR is the crossover constant which take values in the range of $0 \leq CR \leq 1$ and $j_{rand} \in 1, 2, \ldots, D$ is the randomly chosen index. Selection is the key step to choose the vector between the target vector and the trail vector with the aim of creating an individual for the next generation. To keep the population size constant over subsequent generations, the next step of the algorithm calls for selection to the next generation i.e., at $G = G + 1$. The selection operation is described as

$$\mathbf{x}_{i,G+1} = \begin{cases} \mathbf{u}_{i,G}, & \text{if } f(\mathbf{u}_{i,G}) \leq f(\mathbf{x}_{i,G}) \\ \mathbf{x}_{i,G}, & \text{if } f(\mathbf{u}_{i,G}) > f(\mathbf{x}_{i,G}) \end{cases} \tag{30}$$

where $F(\mathbf{x})$ is the function to be minimized. So if the new trail vector yields an equal or lower value of the objective function, it replaces the corresponding target vector in the next generation; otherwise the target is retained in the population. Hence the population either gets better (with respect to the minimization of the objective function) or remains the same in fitness status, but never deteriorates.

4.3 Hybrid Differential Artificial Bee Colony Algorithm

The rationale of proposing a hybrid algorithm is to minimize the inherent drawbacks/ shortcomings of DE and ABCA. Practical experience shows that these algorithms are susceptible to problems like premature or slow convergence [25]. Also, the performance of these algorithms deteriorates with the growth of the dimensionality of the search space and also with increase in complexity of multimodal error. To enjoy the optimum with a good convergence rate DE was combined with ABCA, which might combine their advantages, there by decreasing their disadvantages. Additionally, the search pattern of ABCA combining with DE can enrich the search strategies.

In general, hybridization schemes are broadly categorized into two types: the staged pipelining type hybrid and the other is additional-operator type hybrid [25]. In the first type of hybridization, an optimization process is applied to each and every individual of the population, and the search space is further improved by using the second optimization method. In the second type of hybridization, the optimization algorithm is applied as a standard genetic operator for a given corresponding probability. In the present study, we have employed pipelining type hybrid method because of its advantages. In HDABCA, we apply ABCA to all individuals in the population from which we select the n best vectors based on the fitness values to generate the initial population for local search via DE. There after this population is refined by DE mechanism and the best population obtained through this local search are sent back to ABC. In this way search process continues till stopping criteria is satisfied.

Pseudo Code for HDABCA

1. Initialize the population
2. Send the employed bees to get information about new food sources
3. Send the onlookers based on the probabilities calculated
4. Rank the population based on their objective values obtained
5. Perform local search via Differential Evolution

   Do

      Mutation,
      Crossover
      Selection

6. Memorize the best food sources obtained so far.
7. Repeat this Until stopping criteria is reached

To elucidate the performance of HDABCA we considered four benchmark problems [26]. We are considering four test functions i.e., Rosenbrock (f_1), Ackley (f_2), Schaffer's F6 function (f_3), Goldstien-Price (f_4), of less dimensions , as our problem controller design is of 3-D tuning, this test function values shows the superiority of proposed method on solving low dimensional problems (even though less dimensional the multimodal surface error complexity is high). In this hybrid scheme, once the ABC completes one cycle, 10 best food sources are selected on the basis of fitness and these serves as the initial population for DE. This population is then fine tuned by DE operators' mutation, crossover and selection. The DE process continues for a certain number of generations. After performing several experiments we observed that activating DE for 20 generations gives reasonably good results. All the parameters are summarized in Table 3.

Table 1 provides the comprehensive set of results of mean and std for each algorithm, here Dim represents the dimensions and 2,000 iterations/cycles are considered to be the termination criterion. Considering the stochastic nature, each algorithm was executed 50 times. From this Table 1 it can be observed that under the given parametric settings, the performance of the proposed HDABCA algorithm remained consistently superior to that of basic DE and ABCA for all the benchmark problems. Next section will elucidates about the problem considered and the performance of proposed method in designing the Fractional Controller.

Based on the complexity of function the number of iterations are being changed i.e., for example if the maximum iterations are equal to 1,000 it indicates that maximum iterations are equivalent to 100,000 fitness evaluations. ABCA, DE and the proposed HDABCA are programmed in MATLAB 7.8 on a Pentium IV, 2.2 GHz PC, with 1 MB cache in Windows XP SP3 environment.

5 Design of FOPI controller using HDABCA

The DE, ABCA and HDABCA are used to design the controller parameters K_P, K_I and λ such that the drive exhibits desired response and robust stability as evaluated by the design criteria (Fig. 5). The search space of the controller parameters is three-dimensional and the three dimensions being K_P, K_I, λ. So, in ABCA method for a given dimension the *i*th food source is represented as

$$X_i = [K_P, K_I, \lambda]$$

In case of DE, each parameter vector has three components, i.e., the *j*th population member at *G*th generation may be given as

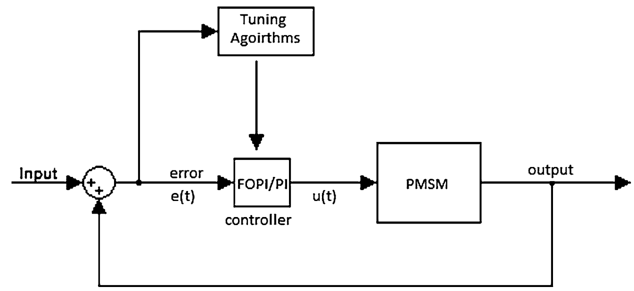

Fig. 5 Tuning of FOPI/PI controller using different methods

$$\vec{X}_{j,G} = [K_P, K_I, \lambda]^T$$

For given set of parameters the tuning is done for each set of algorithm. In the case of Hybrid method the best food sources obtained after each generation of ABCA are set as vectors to DE method and the search process is continued. From the practical consideration of FO-PI (as well as PI) controller, we fixed the following ranges for each of parameter as

- The parametric constants K_P and λ are set between [0,1] and K_I is bounded in range 0 and 10
- ω_l and ω_h in (21)–(25) are set to 10^{-5} and 10^5 rad/s, respectively
- The approximation in (20) is set to $N = 3$

5.1 Design specifications of the drive

Like other motors PMSM drive also consists of certain design criterion to be satisfied, defined or set by the user. Since the basic requirement of a motor is that it should rotate at the desired speed before or on application on load, the steady-state error of the motor should be less than 0.01. The other performance requirement is that motor must accelerate to its steady-state speed as soon as it gets a power supply. In this case, we want it to have a settling time of 0.6 s. Since a speed faster than the reference may damage the equipment, we want to have an overshoot of less than 2 %. Simulation is done for time $T = 1$ s under a load torque of 5-Nm with a reference speed of 500 rpm.

Table 1 Average and the standard deviation (SD) of the best-of-run solution for 50 independent runs

Fun	Dim	DE	ABC	HDABCA
f_1	30	1.2407E+00 (2.272E+00)	2.2113E+00 (3.8982E+00)	0.109 (5.646−17)
f_2	30	7.3131E+00 (2.0813E+00)	1.457E−12 (7.2509E−13)	4.4408E−15 (0)
f_3	2	1.042E−04 (5.708E−04)	2.4785E−03 (3.8469−003)	0 (0)
f_4	2	3.9E+00 (4.929E+00)	3.4052E+00 (7.4341E−01)	3 (1.8067E−15)

5.2 Experimental settings of PMSM

This section briefs about the parameters considered for Permanent Magnet Synchronous Motor. The following are experimental settings and are summarized in Table 2.

5.3 Parameter settings of competing algorithms

The proposed design method has been extensively compared with two state-of-the-art design methods for FO-PI controllers based on the DE and ABCA. It was also compared with the traditional derivative based Gradient or Steepest descent method. A termination criterion of 100 iterations/cycles had been considered for tuning the FO-PI controller. No hand tuning of parameters has been allowed in any case to make comparisons fair enough. Various parameters involved in these methods are summarized in Table 3.

6 Results and discussions

6.1 Integer order PI controller

In this section the performance of PMSM controlled by IO-PI is investigated for different tuning algorithms. Figure 6 shows the speed response of PMSM before the application of load and Fig. 7 shows the response after the load is applied (i.e., after $T = 0.5$ s). It is very clear from the plots and the comparisons Table 5 that performance of IO-PI controller is increased, when designed compatibly by using Hybrid Differential Artificial Bee Colony Algorithm. Also it can be noticed that the Peak over shoot, settling time and steady state error is considerably decreased (Table 4).

6.2 Fractional order PI controller

After the extensive study of PI controller, we analyzed if the performance can be further improved. For this purpose we employed Fractional Order Controllers. This section provides the study of PMSM controlled by Fractional order

Table 2 Parameter setting of PMSM drive

Variable	Actual implication	Value	Units
r_s	Stator resistance	2.0	Ω
L_{sd}	*d* axis inductance	2.419	mH
L_{sq}	*q* axis inductance	2.419	mH
J	Moment of inertia	0.00344638	kg-m^2
λ_m	Magnet mutual flux	0.27645	V/rad/s
B	Damping coefficient	0.0027715	Nm/rad/s
P	Number of poles	8	–

Table 3 Algorithmic parameters

Parameters	DE	ABCA	HDABCA
Popsize	20	20	20
F	0.5	Not required	0.5
CR	0.8	Not required	0.8
limit	Not required	$n_e{*}D$	$n_e{*}D$

Table 4 Comparisons step response of PI controller using different Methods

Method	K_P	K_I	*po* (%)	t_r (s)	t_s (s)	e_{ss}
Gradient	0.0672	7.8764	33.3198	0.0216	0.6627	0.0301
DE	0.1027	8.5368	18.3619	0.0207	0.6134	0.0266
ABCA	0.1783	8.9728	14.6448	0.0199	0.6086	0.0273
HDABCA	0.2896	9.5124	**3.6757**	**0.0152**	**0.5927**	**0.0059**

Table 5 Comparisons step response of FO-PI controller using different methods

Method	K_P	K_I	λ	*po*(%)	t_r (s)	t_s (s)	e_{ss}
Gradient	0.1142	6.731	0.2	15.309	0.0146	0.5623	0.00792
DE	0.1815	7.9183	0.5	9.0316	0.0185	0.5718	0.00909
ABCA	0.2406	8.239	0.5	5.0251	0.0202	0.5683	0.00734
HDABCA	0.3592	8.8691	0.7	**0.0210**	**0.0033**	**0.5521**	**0.00532**

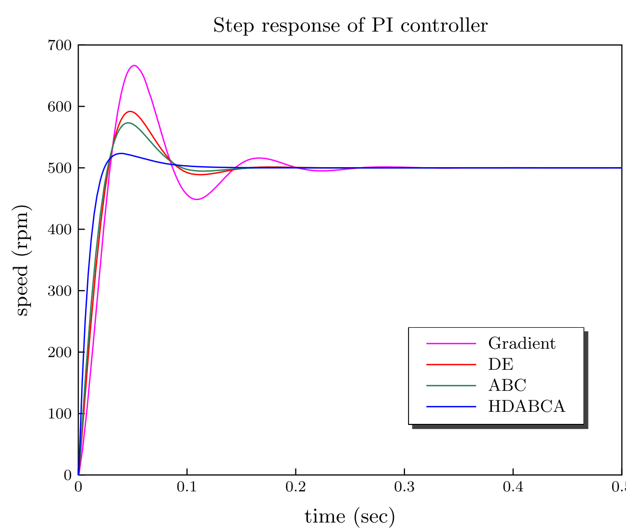

Fig. 6 Step response of PMSM controlled by IO-PI before load

Proportional Controller (FO-PI), for different proposed tuning algorithms. Figure 8 shows the speed response of PMSM before the application of load and Fig. 9 shows the response after the load is applied (i.e., after $T = 0.5$ s). From the obtained responses and observations it is eminent that FO-PI controller designed with hybrid method performs better than conventional steepest descent method and

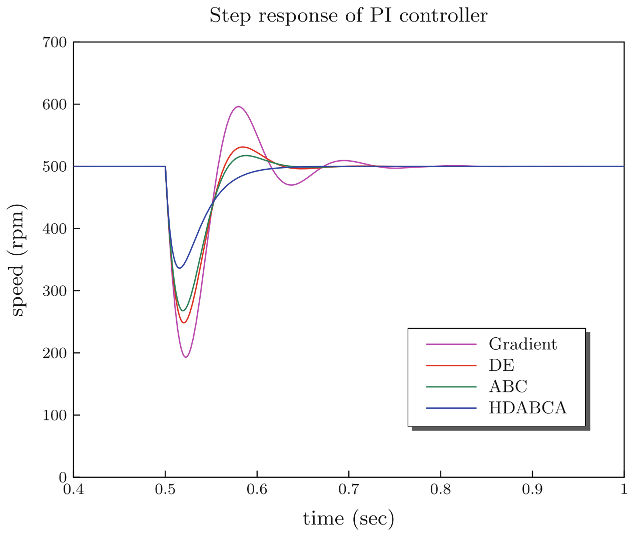

Fig. 7 Step response of PMSM controlled by IO-PI after load

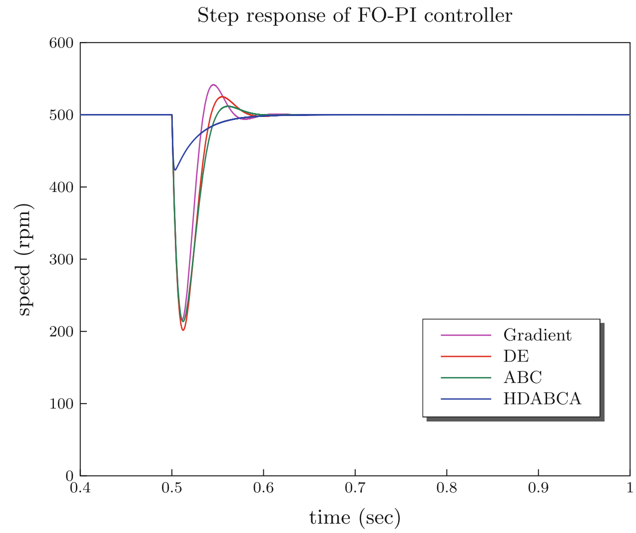

Fig. 9 Step response of PMSM controlled by FO-PI after load

the basic ABCA and DE algorithms. Almost every design constraint is satisfied with the proposed method and especially the percentage overshoot is being considerably reduced.

6.3 Best IO-PI versus FO-PI

In this section we compared the performance of best IO-PI controller and FO-PI controller, to resolve ambiguity, and to choose which of the controller, for an optimum performance. From the extensive simulations and comparisons it is quite interesting to note that PMSM controlled PI controller when designed with proposed method performs better than the remaining methods, but still there are unsatisfactory results in terms of peak overshoot and settling time (with in a given bound). This is easily resolved

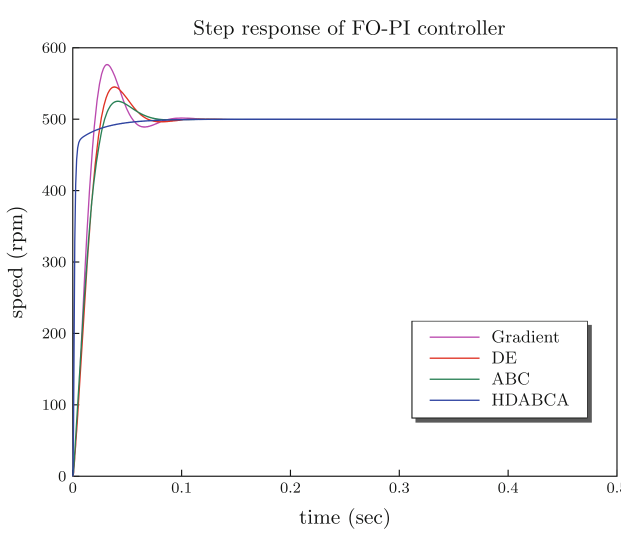

Fig. 8 Step response of PMSM controlled by FO-PI before load

Table 6 Best IO-PI versus FO-PI

Controller	K_P	K_I	λ	po (%)	t_r (s)	t_s (s)	e_{ss}
IO-PI	0.2896	9.5124	1	3.6757	0.0152	0.5927	0.0059
FO-PI	0.3592	8.8691	0.7	**0.0210**	**0.0033**	**0.5521**	**0.00532**

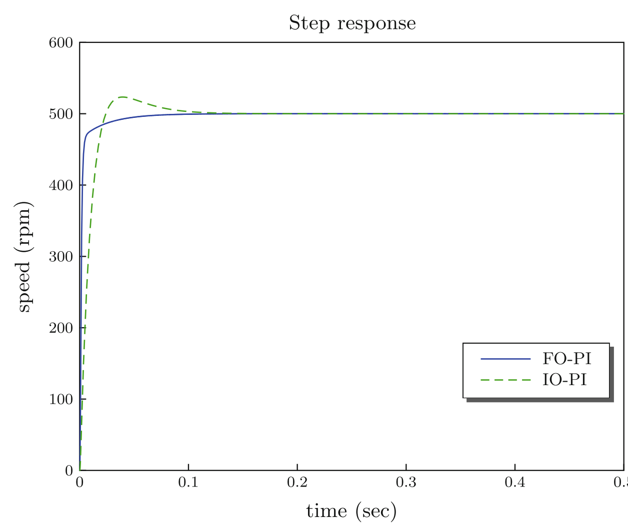

Fig. 10 Step response of PMSM controlled by IO-PI and FO-PI before load

by using the fractional order control due to an additional constraint λ, from Table 6 we can see that not only peak overshoot is decreased but also rises time also got improved. Figures 10 and 11 shows the step response of IO-PI and FO-PI controlled PMSM. The convergence characteristics of proposed HDABCA along with ABC and DE, in optimizing ITAE function (for FOPI controller) is shown in Fig 12.

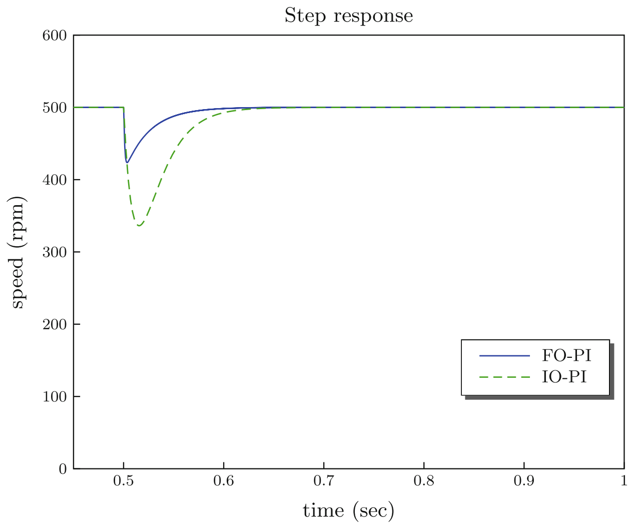

Fig. 11 Step response of PMSM controlled by IO-PI and FO-PI after load

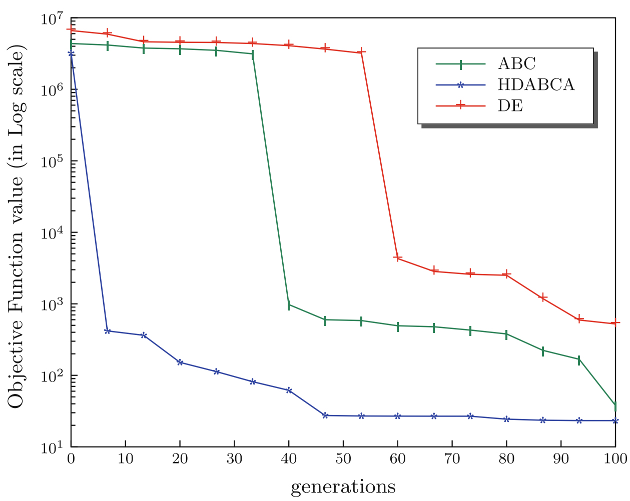

Fig. 12 Convergence characteristics for an FO-PI controlled PMSM

7 Conclusions

In this paper, an intelligent optimization method, HDABCA (a hybrid version of DE and ABCA), for designing FO-PI Speed controller in a PMSM drive is presented. Simulations and comparisons shown in the paper clearly indicate that a properly designed and implemented FO-PI controller provides better results than that of traditional integer-order PI controller. It can be seen that the performance of PMSM controlled by best tuned FO-PI controller is quite satisfactory in comparison to IO-PI controller. In this application, for FO-PI controller, Oustaloup approximation method was used for digital realization purposes. The proposed method has been shown to outperform a state-of-the-art version of the DE algorithm and ABCA method especially for the fractional-order controller. Our further research would include the performance and analysis fractional order controller in a sensor less motor.

References

1. Bose BK (1993) Power electronics and motion control-technology status and recent trends. IEEE Trans Ind Appl 29(5):902–909
2. Vas P (1998) Sensor less Vector and Direct Torque Control, 1st edn, Oxford University Press, Oxford
3. Leonhard W (1986) Microcomputer control of high dynamic performance ac drives—a survey. Automatica 22(1):1–19
4. Slotine JJ, Li W (1991). Applied nonlinear control. Pearson Education, Englewood Cliffs
5. Astrom KJ, Hugglund T (2001) The future of PID control. Control Eng Pract 9:1163–1175
6. Podlubny I (1994) Fractional-order systems and fractional-order controllers. The Academy of Sciences Institute of Experimental Physics, UEF-03-94, Kosice, Slovak Republic
7. Goldberg D (1986) Genetic algorithms in search optimization and machine learning. Addision Wesley Publishing Company, Massachutes
8. Jatoth RK, Rajasekhar A (2010) Adaptive bacterial foraging optimization based tuning of optimal PI speed controller for PMSM drive. International Conference on Contemporary Computing, IC3-2010, pp 588–599
9. Lin C-M, Li M-C, Ting A-B, lin M-H (2011) A robust self-learning PID control system design for nonlinear systems using a particle swarm optimization algorithm. Int J Mach Learn Cybern 2(4):225–234
10. Wang X, He Y, Dong L, Zhao H (2011) Particle Swarm optimization for determining fuzzy measures from data. Inf Sci 181(19):4230–4252
11. Price K, Storn R (1995) Differential evolution-a simple and efficient adaptive scheme for global optimization over continous spaces, Technical Report, International Computer Science Institue, Berkley
12. Karaboga D (2005) An idea based on bee swarm for numerical optimization. Technical Report, TR-06, Erciyes University Engineering Faculty, Computer Engineering Department
13. Das S, Biswas A, Abraham A, Dasgupta S (2009) Design of fractional order $PI^{\lambda}D^{\mu}$ controllers with an improved differential evolution. Eng Appl Artif Intell 22(2):343–350
14. Sabat S, Udgata S, Abraham A (2010) Artificial Bee Colony Algorithm for small signal model parameter extraction of MESFET. Eng Appl Artif Intell 23(5):689–694
15. Rajasekhar A, Das S, Suganthan P (2012) Design of fractional order controller for a servohydraulic positioning system with micro Artificial Bee Colony Algorithm. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp 1–8
16. Sebastian T, Slemon GR (1986) Transient modeling and performance of variable—speed permanent-magnet motors. IEEE Trans Ind Appl 25(1):101–106
17. Pillay P, Krishnan R (1989) Modeling, simulation, and analysis of permanent-magnet motor drives, part i: the permanent-magnet synchronous motor drive. IEEE Trans Ind Appl 25(2):265–273
18. Zamani M, Ghartemani MK, Sadati N, Parniani M (2009) Design of a fractional order PID controller for an AVR using particle swarm optimization. Control Eng Pract 17(12):1380–1387
19. Oldham KB, Spanier J (1974) The fractional calculus. Academic Press, New York
20. Petras I (1999) The fractional order controllers: methods for their synthesis and application. J Electr Eng 50(910):284–288.

21. Podlubny I (1999) Fractional-order systems and $PI^{\lambda}D^{\mu}$ -controllers. IEEE Trans Autom Control 44(1):208–214
22. Chen Y., Xue D., Dou H. (2004). Fractional calculus and biomimetic control. In: Proceedings of the First IEEE International Conference on Robotics and Biomimetics (RoBio04), robio2004, pp 347, Shengyang, China
23. Krohling R A, Rey J P (2001) Design of optimal disturbance rejection PID controller using genetic algorithms. IEEE Trans Evol Comput 5(1):78–82
24. Price K, Storn R, Lampinen J (2005) Differential evolution— a practical approach to global optimization. Springer, New York
25. Lampinen J, Zelinka I (2000) On stagnation of the differential evolution algorithm. In: Proceedings of the MENDEL, 6th international mendel conference on soft computing, Brno, Czech Republic, pp 76–83
26. Dasguta S, Das S, Abraham A, Biswas A (2009) Adaptive computational chemotaxis in bacterial foraging optimization: an analysis. IEEE Trans Evol Comput 13(4):919–941
27. Rajasekhar A, Pant M, Abraham A (2011) A Hybrid Differential Artificial Bee Algorithm based tuning of fractional order controller for PMSM drive. Third World Congress on Nature and Biologically Inspired Computing, NaBIC-2011, pp 19–21