AN ENERGY EFFICIENT CONTROL STRATEGY FOR INDUCTION MACHINES BASED ON ADVANCED PARTICLE SWARM OPTIMISATION ALGORITHMS

Duy C. Huynh[1] and Matthew W. Dunnigan[2]
[1]Department of Power System Engineering, University of Technology,
Vietnam National University of Hochiminh City, Hochiminh City, Vietnam
[2]Department of Electrical, Electronic and Computer Engineering, Heriot-Watt University,
Edinburgh, United Kingdom

ABSTRACT

This paper proposes an energy efficient control strategy for an induction machine (IM) based on two advanced particle swarm optimisation (PSO) algorithms. Two advanced PSO algorithms, known as the dynamic particle swarm optimisation (Dynamic PSO) and the chaos particle swarm optimisation (Chaos PSO) algorithms modify the algorithm parameters to improve the performance of the standard PSO algorithm. These parameters are used to determine an optimal rotor flux reference for loss model-based energy efficient control of an IM. There is also a comparison of the results obtained when using a GA, standard PSO, dynamic PSO and chaos PSO algorithms. The comparison confirms the validity and effectiveness of the proposed energy efficient control strategy.

KEYWORDS: *Energy Efficient Control, Induction Machines, Particle Swarm Optimisation Algorithm*

I. INTRODUCTION

Energy efficient control of the induction machine (IM) has received significant attention in recent years because of concerns regarding energy saving and environmental pollution reduction. Basically, the IM operational efficiency is high for rated conditions of the load torque, speed and flux. Nevertheless, IM drive systems usually operate at light loads most of the time. In this case, if the rated flux is maintained at light loads, the core loss will increase dramatically. This results in poor IM efficiency. Various approaches have been researched to enhance the IM efficiency at light loads. Two basic control approaches, known as model-based control and search control have been introduced. The model-based control approach uses an IM loss model to define an optimal flux for each operational point at a given load torque and machine speed. This approach has a fast response time. However, it is not robust to IM parameter variations. A neural network [1-6], a genetic algorithm [7-8] and a particle swarm optimisation algorithm [9] have allowed an optimal flux level to be defined for energy efficient control using the IM loss model. In the model-based control approach, the IM loss model is usually formed by the IM loss components such as the stator and rotor copper losses, core loss, stray loss and mechanical losses [3-5] and [8-9]. The search control approach is based on a search of optimal flux levels which ensure minimization of the IM measured input power for a given load torque and machine speed. It can be deduced that this approach is insensitive to IM parameter variations and does not require a priori knowledge of the IM parameters. Nevertheless, the response for obtaining an optimal flux value is slow. Additionally, input power measurement noise can affect the algorithm performance. A fuzzy logic [10-15] and a golden section technique [16] have been applied for this control strategy. It is obvious that there are always disadvantages in the model-based control and search control approaches. This is why hybrid controllers [17-23] have been recently examined for energy efficient control of the IM. These are a combination of the model-based control and search control approaches. By using hybrid controllers, the energy efficient control strategy always remains optimal. Nevertheless, it can be deduced that the structure of these controllers is complex.

This paper proposes a loss model-based energy efficient control strategy for the IM using an optimal rotor flux reference which is determined using two advanced particle swarm optimisation (PSO) algorithms, known as the dynamic particle swarm optimisation (Dynamic PSO) and the chaos particle swarm optimisation (Chaos PSO) algorithms. Simulations and comparisons are performed to confirm the effectiveness and benefit of the proposed energy efficient control strategy.

The remainder of this paper is organized as follows. An energy efficient control strategy using an optimal rotor flux reference is presented in Section 2. The new application of the dynamic PSO and chaos PSO algorithms is proposed in Section 3. The simulation results then follow to confirm the validity of the proposed techniques in Section 4. Finally, the advantages of the new techniques are summarised through comparison with the basic PSO and genetic algorithms.

II. ENERGY EFFICIENT CONTROL OF AN INDUCTION MACHINE

In the model-based control approach, most of the previous energy efficient control strategies were based on the model of the IM loss components which are the stator and rotor copper losses, core loss, stray loss and mechanical losses. This paper introduces a loss model for energy efficient control of the IM which is more general and simpler than others. In this case, energy efficient control is considered in the steady-state and d-axis indirect rotor flux-oriented control conditions. Thus, the IM mathematical model is described as follows [24].

$$v_{qs} = R_s i_{qs} + \omega_e L_s i_{ds} \tag{1}$$

$$v_{ds} = R_s i_{ds} + \omega_e \left(\frac{L_m^2 - L_s L_r}{L_r} \right) i_{qs} \tag{2}$$

$$i_{qs} = \frac{1}{R_r} \frac{L_r}{L_m} \left(\omega_e - \omega_r \right) \psi_{dr} \tag{3}$$

$$i_{ds} = \frac{1}{L_m} \psi_{dr} \tag{4}$$

$$T_e = \frac{3}{2} \frac{p}{2} \frac{L_m}{L_r} \psi_{dr} i_{qs} \tag{5}$$

where

v_{ds}, v_{qs}, i_{ds} and i_{qs} are the d-q axis stator voltages and currents.

R_s, R_r, L_s, L_r and L_m are the stator and rotor resistances, stator and rotor inductances and magnetizing inductance.

ω_e is the synchronous speed.

ω_r and ω_m are the rotor electrical and mechanical speeds.

ψ_{dr} is the d-axis rotor flux.

T_e is the electrical torque.

p is the number of poles.

From (3) and (5), the IM synchronous speed is given by:

$$\omega_e = \omega_r + \frac{4}{3} \frac{R_r T_e}{p} \frac{1}{\psi_{dr}^2} \tag{6}$$

Substituting (3)-(4) and (6) into (1)-(2), the d-q axis stator voltages become:

$$v_{ds} = \frac{R_s}{L_m}\psi_{dr} + \frac{4}{3}\frac{T_e}{p}\left(\frac{L_m^2 - L_s L_r}{L_m}\right)\omega_r \frac{1}{\psi_{dr}} + \frac{16}{9}\frac{T_e^2 R_r}{p^2}\left(\frac{L_m^2 - L_s L_r}{L_m}\right)\frac{1}{\psi_{dr}^3} \tag{7}$$

$$v_{qs} = \frac{4}{3}\frac{T_e}{p}\left(\frac{R_r L_m + R_s L_s}{L_m}\right)\frac{1}{\psi_{dr}} + \frac{L_s}{L_m}\omega_r \psi_{dr} \tag{8}$$

From (3)-(4) and (6)-(8), assuming that the stator and rotor inductances are the same value, the input power of the IM is then given as follows:

$$P_{in} = v_{qs} i_{qs} + v_{ds} i_{ds} = \frac{R_s}{L_m^2}\psi_{dr}^2 + \frac{16}{9}\frac{T_e^2}{p^2}\left(\frac{R_r L_m^2 + R_s L_s^2}{L_m^2}\right)\frac{1}{\psi_{dr}^2} + \frac{4}{3}\frac{T_e}{p}\omega_r \tag{9}$$

In addition, the output power of the IM is described as follows:

$$P_{out} = \omega_m T_e = \frac{2}{p}\omega_r T_e \tag{10}$$

Combining (9) and (10), the total IM loss is:

$$\Delta P = P_{in} - P_{out} = \frac{R_s}{L_m^2}\psi_{dr}^2 + \frac{16}{9}\frac{T_e^2}{p^2}\left(\frac{R_r L_m^2 + R_s L_s^2}{L_m^2}\right)\frac{1}{\psi_{dr}^2} - \frac{2}{3}\frac{T_e}{p}\omega_r \tag{11}$$

The IM efficiency can be improved by minimizing the total IM loss which is dominated by the stator and rotor copper losses and core loss. The stator and rotor copper losses are reduced by decreasing the stator and rotor currents respectively which results in increased IM flux. As a consequence, the core loss is then increased. Obviously, there is a conflict between the copper losses and core loss. When the copper losses are decreased, the core loss is increased [25]. Nevertheless, there is an optimal IM flux at which the total IM loss is minimized for a given load torque and machine speed [24]. As a result, the solution for energy efficient control of the IM is to find the optimal IM flux reference during operation. This is based on the IM loss model defined in (11). In order to solve this problem, the dynamic PSO and chaos PSO algorithms are two of the relatively new population-based stochastic optimisation algorithms, which are proposed to obtain an optimal IM flux reference for energy efficient control of the IM. The algorithms are presented in detail in the next section.

III. ENERGY EFFICIENT CONTROL OF AN INDUCTION MACHINE USING ADVANCED PARTICLE SWARM OPTIMISATION ALGORITHMS

The standard PSO algorithm is reviewed in part 3.1 followed by descriptions of two advanced PSO algorithms: the dynamic PSO and chaos PSO algorithms in parts 3.2 and 3.3 of this section respectively.

3.1. Standard Particle Swarm Optimisation Algorithm

The PSO algorithm is a population-based stochastic optimisation method which was developed by Eberhart and Kennedy in 1995 [26]. The algorithm was inspired by the social behaviors of bird flocks, colonies of insects, schools of fishes and herds of animals. The algorithm starts by initializing a population of random solutions called particles and searches for optima by updating generations through the following velocity and position update equations.

The velocity update equation:

$$\boldsymbol{v_i}(k+1) = w\boldsymbol{v_i}(k) + c_1 r_1(\boldsymbol{pbest_i}(k) - \boldsymbol{x_i}(k)) + c_2 r_2(\boldsymbol{gbest}(k) - \boldsymbol{x_i}(k)) \tag{12}$$

The position update equation:

$$\boldsymbol{x_i}(k+1) = \boldsymbol{x_i}(k) + \boldsymbol{v_i}(k+1) \tag{13}$$

where

$\boldsymbol{v_i}(k)$ is the *k*th current velocity of the *i*th particle.

$\boldsymbol{x_i}(k)$ is the *k*th current position of the *i*th particle.

k is the *k*th current iteration of the algorithm, $1 \le k \le n$.

n is the predefined maximum iteration number.

i is the *i*th particle of the swarm, $1 \le i \le M$.

M is the particle number of the swarm.

Usually, $\boldsymbol{v}_i$ is clamped in the range [$-v_{max}$, v_{max}] to reduce the likelihood that a particle might leave the search space. In case of this, if the search space is defined by the bounds [$-x_{max}$, x_{max}] then the v_{max} value will be typically set so that $v_{\max} = mx_{\max}$, where $0.1 \le m \le 1.0$ [27].

$\boldsymbol{pbest_i}(k)$ is the best position found by the *i*th particle (personal best).

$\boldsymbol{gbest}(k)$ is the best position found by a swarm (global best, best of the personal bests).

c_1 and c_2 are the cognitive and social parameters. The parameter, c_2 regulates the step size in the direction of a global best particle and the c_1 regulates the step size in the direction of a personal best position of that particle, c_1 and $c_2 \in [0, 2]$. With large cognitive and small social parameters at the beginning, particles are allowed to move around a wider search space instead of moving towards a population best. On the other hand, with small cognitive and large social parameters, particles are allowed to converge to the global optimum in the latter part of optimisation [28].

r_1 and r_2 are two independent random sequences which are used to influence the stochastic nature of the algorithm, $r_1 \in U(0, 1)$ and $r_2 \in U(0, 1)$.

w is the inertia weight [29].

The velocity update equation of the particle is considered as three parts: the first part is the previous velocity of the particle, $w\boldsymbol{v}_i(k)$; the second and the third parts, $c_1r_1(\boldsymbol{pbest}_i(k)-\boldsymbol{x}_i(k))$ and $c_2r_2(\boldsymbol{gbest}(k)-\boldsymbol{x}_i(k))$, contribute to the particle velocity change.

Without the first part of the velocity update equation, the particles' velocities are only determined by their current and best history positions and the PSO algorithm search process is similar to a local search algorithm. Thus, the particles tend to move towards the same position and the final solution depends heavily on the initial population. The PSO algorithm only finds out the final solution when the initial search space includes the global optimum. By adding the first part, the particles have a tendency to expand the search space and explore the new area. Because of this, the PSO algorithm becomes a global search algorithm. Nevertheless, for each problem, there is always a different trade-off between the local and global search abilities. This is why the inertia weight is used in the first part [29]. This value was set to 1 in the original PSO algorithm [26]. Shi and Eberhart investigated the effect of *w* values in the range [0, 1.4] as well as in a linear time-varying domain. Their results indicated that choosing $w \in [0.9, 1.2]$ results in a faster convergence [29]. A larger inertia weight facilitates a global exploration and a smaller inertia weight tends to facilitate a local exploration [30]. Therefore, careful choice of the inertia weight *w* during the evolution process of the PSO algorithm is necessary. This improves the convergence capability and search performance of the algorithm.

The two remaining parts of the velocity update equation also play an important role in updating the new velocities of the particles. The term $(\boldsymbol{pbest}_i(k)-\boldsymbol{x}_i(k))$ is the distance of its own best position from its current position whereas the term $(\boldsymbol{gbest}(k)-\boldsymbol{x}_i(k))$ is the distance of the best position in the swarm from its current position. Without the second and third parts, the particles will keep their current speed in the same direction until they hit the boundary [29]. This affects the algorithm performance during the evolution process.

Eventually, the particle flies towards a new position according to the position update equation (13) using the previous position and new velocity of the particle.

The performance of each particle is based on a predefined fitness function which is related to the particular application (11).

In this application, the particles represent the rotor flux reference of the IM. The *i*th particle position and velocity are limited as follows:

$$\psi_{dri(min)} \le \psi_{dri} \le \psi_{dri(max)} \tag{14}$$

and

$$v_{\psi_{dri(\min)}} \le v_{\psi_{dri}} \le v_{\psi_{dri(\max)}} \tag{15}$$

In this application, the acceleration coefficients, c_1 and c_2, are set to 2. The inertia weight, w, is set to 0.9. The two independent random sequences, r_1 and r_2, are uniformly distributed in U(0, 1).

The best position of the *i*th particle, { $\boldsymbol{pbest}_{\psi_{dri}}(k)$ }, and the best position over the swarm, { $\boldsymbol{gbest}_{\psi_{dr}}(k)$ }, are obtained at each *k*th iteration using the fitness function (11).

The update mechanism of the personal { $\boldsymbol{pbest}_{\psi_{dri}}(k)$ } and global { $\boldsymbol{gbest}_{\psi_{dr}}(k)$ } bests is described as follows:

In case the fitness value of the *i*th particle at the *k*th iteration step is better than that of the { $\boldsymbol{pbest}_{\psi_{dri}}(k-1)$ } at the (*k*-1)th iteration step then the *i*th particle will be set to { $\boldsymbol{pbest}_{\psi_{dri}}(k)$ } whereas if the fitness value of the *i*th particle at the *k*th iteration step is better than that of { $\boldsymbol{gbest}_{\psi_{dr}}(k-1)$ } then the global best, { $\boldsymbol{gbest}_{\psi_{dr}}(k)$ }, will be updated corresponding to the *i*th particle at the *k*th iteration step.

The evolution process of the standard PSO algorithm is implemented according to the position and velocity update equations, (13) and (12), respectively.

Eventually, the standard PSO algorithm stops at the *n*th maximum iteration number and the optimal rotor flux reference is obtained as follows.

$$\psi_{dr\_optimal} = gbest_{\psi_{dr}}(n) \tag{16}$$

3.2. Dynamic Particle Swarm Optimisation Algorithm

A dynamic PSO algorithm is one of the standard PSO algorithm variants which was introduced in [28] with time-varying cognitive and social parameters. For most of the population-based optimisation techniques, it is desirable to encourage the individuals to wander through the entire search space without clustering around local optima during the early stages of the optimisation, as well as being important to enhance convergence towards the global optimum during the latter stages. The second part of the velocity update equation (12) is known as the cognitive component which represents the personal thinking of each particle. The cognitive component encourages the particles to move towards their own best positions whereas the third part of the velocity update equation is known as the social component which represents the collaborative effect of the particles in the global optimal solution search. The social component always pulls the particles towards the global best particle [28]. Thus, it is obvious that the cognitive and social parameters in the velocity update equation are two of the parameters which support the algorithm to satisfy the requirements of enhancing the performance in the early and latter stages. Proper control of these two parameters is important to find the optimal solution accurately and efficiently. Using the modification of the cognitive and social parameters, the algorithm improves the global search capability of the particles in the early stage of the optimisation process and then directs particles to the global optimum at the end stage so that the convergence capability of the search process is enhanced. To achieve this, large cognitive and small social parameters are used at the beginning and small cognitive and large social parameters are used at the latter stage. The mathematical representation of this modification is given as follows [28]:

$$\boldsymbol{v}_i(k+1) = w\boldsymbol{v}_i(k) + c_1(k)r_1(\boldsymbol{pbest}_i(k) - \boldsymbol{x}_i(k)) + c_2(k)r_2(\boldsymbol{gbest}(k) - \boldsymbol{x}_i(k)), \quad (17)$$

$$1 \le i \le M \text{ and } 1 \le k \le n$$

where

$$c_1(k) = (c_{1final} - c_{1initial})\frac{k}{n} + c_{1initial} \quad (18)$$

$$c_2(k) = (c_{2final} - c_{2initial})\frac{k}{n} + c_{2initial} \quad (19)$$

$c_1(k)$ and $c_2(k)$ are the time-varying cognitive and social parameters.

$c_{1initial}$ and c_{1final} are the initial and final values respectively of the cognitive parameter.

$c_{2initial}$ and c_{2final} are the initial and final values respectively of the social parameter.

The dynamic PSO algorithm is applied for energy efficient control of the IM where the position and velocity of the *i*th particle are updated using (13) and (17) respectively. The velocity update equation uses the time-varying cognitive and social parameters.

In this application, the parameter $c_1(k)$ is set to decrease linearly with $c_{1initial} = 2.5$ and $c_{1final} = 0.5$ during a run whereas the parameter $c_2(k)$ is set to increase linearly $c_{2initial} = 0.5$ and $c_{2final} = 2.5$.

Thus, the cognitive parameter is large and the social parameter is small at the beginning. This enhances the global search capability in the early part of the optimisation process. Then, the cognitive parameter is decreased linearly and the social parameter is increased linearly until at the end of the search, the particles are encouraged to converge towards the global optimum with small cognitive and large social parameters. This modification improves the evolution process performance and overcomes premature convergence of the standard PSO algorithm.

Additionally, the particles also represent the rotor flux reference of the IM with the limitations of the *i*th particle position and velocity as in (14) and (15).

The inertia weight, w, is set to 0.9. The two independent random sequences, r_1 and r_2, are uniformly distributed in U(0, 1).

The evolution process of the dynamic PSO algorithm is implemented according to the position and velocity update equations, (13) and (17), respectively.

Eventually, the dynamic PSO algorithm stops at the *n*th maximum iteration number and the optimal rotor flux reference is obtained by (16).

3.3. Chaos Particle Swarm Optimisation Algorithm

In addition to the dynamic PSO algorithm, this paper also proposes another novel application of a chaos PSO algorithm for energy efficient control of the IM which is also more efficient than the standard PSO algorithm. The chaos PSO algorithm is a combination between the standard PSO algorithm and a chaotic map which was presented in [30-34].

Chaos is a common phenomenon in non-linear systems, which includes infinite unstable period motions. It is a stochastic and unpredictable process in a deterministic non-linear system.

A chaotic map is a discrete-time dynamical system [30] which is given as follows:

$$x_k = f(x_{(k-1)}) \quad (20)$$

where $x_k \in (0, 1)$, $k = 1, 2, \ldots$

The sequences are generated by using one of the chaotic maps known as chaotic sequences. These sequences have the characteristics of the chaotic map such as randomness, ergodicity and regularity, so that no state is repeated. The chaotic sequences have been recently considered as sources of random sequences which can be adopted instead of normally generated random sequences.

For the standard PSO algorithm, one of its main disadvantages is premature convergence, especially in local optima problems. Thus, in order to overcome this, the algorithm parameter sequences with a randomness-based choice are substituted by the chaotic sequences which are generated from a chaotic map. In this case, the chaotic sequences are obviously an appropriate tool to support the standard PSO algorithm so that it avoids getting stuck in a local optimum during the search process and overcomes the premature convergence phenomenon present in the standard PSO algorithm. There are many chaotic maps which have been introduced and used to improve the standard PSO algorithm [30]. Amongst them, the logistic map is one of the simplest and easiest maps to employ in the chaos PSO algorithm for energy efficient control of the IM.

A logistic map is given as follows:

$$X_k = aX_{(k-1)}\left(1 - X_{(k-1)}\right),\ k = 1, 2, \ldots \tag{21}$$

where

X_k is the kth chaotic number, $X_k \in (0, 1)$ with the following initial conditions.

X_0 is a random number in the interval of (0, 1) and $X_0 \notin \{0.0, 0.25, 0.5, 0.75, 1.0\}$.

a is the control parameter, usually set to 4 in the experiments [30].

In this application, the particles represent the rotor flux reference of the IM. Each particle has its position and velocity. The logistic map is used for initializing the position $\{\boldsymbol{\psi}_{dri}\}$ and velocity $\{\boldsymbol{v}_{\psi_{dri}}\}$ of the ith particle described as follows:

$$\psi_{dri}(1) = b\psi_{dr(i-1)}(1)\left(1 - \psi_{dr(i-1)}(1)\right),\ 1 \le i \le M \tag{22}$$

where

$\psi_{dr0}(1)$ is an initial value to produce the first particle position at the first iteration. It is a random number in the interval of (0, 1) and $\psi_{dr0}(1) \notin \{0.0, 0.25, 0.5, 0.75, 1.0\}$.

$$v_{\psi_{dri}}(1) = bv_{\psi_{dr(i-1)}}(1)\left(1 - v_{\psi_{dr(i-1)}}(1)\right),\ 1 \le i \le M \tag{23}$$

where $v_{\psi_{dr0}}(1)$ is an initial value to produce the first particle velocity at the first iteration. It is a random number in the interval of (0, 1) with $v_{\psi_{dr0}}(1) \notin \{0.0, 0.25, 0.5, 0.75, 1.0\}$.

The ith particle position and velocity are also limited by (14) and (15).

In addition, the chaotic inertia weight in the chaos PSO algorithm is:

$$w_k = bw_{(k-1)}\left(1 - w_{(k-1)}\right),\ 1 \le k \le n \tag{24}$$

where

w_k is the kth chaotic inertia weight, $w_k \in (0, 1)$ has the following initial conditions.

w_0 is a random number in the interval of (0, 1) and $w_0 \notin \{0.0, 0.25, 0.5, 0.75, 1.0\}$.

Moreover, the two independent chaotic random sequences in the chaos PSO algorithm are:

$$r_k^1 = br_{(k-1)}^1\left(1 - r_{(k-1)}^1\right),\ 1 \le k \le n \tag{25}$$

$$r_k^2 = br_{(k-1)}^2\left(1 - r_{(k-1)}^2\right),\ 1 \le k \le n \tag{26}$$

where

r^1_k and r^2_k are two kth independent chaotic random sequences, r^1_k and $r^2_k \in (0, 1)$ have the following initial conditions: r^1_0 and r^2_0 are random numbers in the interval of (0, 1) and r^1_0 and $r^2_0 \notin \{0.0, 0.25, 0.5, 0.75, 1.0\}$.

Then, the velocity update equation of the standard PSO algorithm is re-written as follows:

$$\boldsymbol{v_i}(k+1) = w_k\boldsymbol{v_i}(k) + c_1r_k^1\left(\boldsymbol{pbest_i}(k) - \boldsymbol{x_i}(k)\right) + c_2r_k^2\left(\boldsymbol{gbest}(k) - \boldsymbol{x_i}(k)\right), \tag{27}$$

$$1 \le i \le M \text{ and } 1 \le k \le n$$

where

w_k , r_k^1 and r_k^2 are the logistic maps.

In this case, the cognitive and social parameters, c_1 and c_2 are set to 2.

The evolution process of the chaos PSO algorithm is implemented according to the position and velocity update equations, (13) and (27) respectively.

Eventually, the chaos PSO algorithm will stop at the *n*th maximum iteration number and the optimal rotor flux reference will be obtained by (16).

The flow chart for energy efficient control of the IM using the standard PSO, dynamic PSO and chaos PSO algorithms is shown in Figure 1.

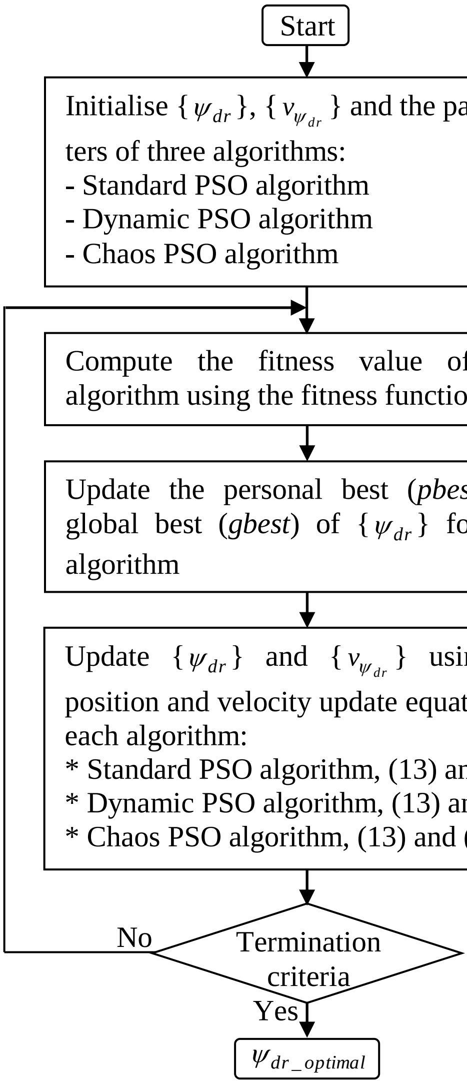

Figure 1. Flow chart for energy efficient control of the IM using the standard PSO, dynamic PSO and chaos PSO algorithms.

IV. SIMULATION RESULTS

Simulations are performed using MATLAB/SIMULINK software for energy efficient control of the 3 Hp IM, fed by a voltage source inverter. The specifications and parameters of the simulated IM are in Table 1. The standard PSO, dynamic PSO and chaos PSO algorithms are applied for energy efficient control of the IM in which the particle number of a generation is set to 50 and the maximum iteration number is set to 100.

Table 1. IM specifications and parameters.

Number of phases	3
Connection	Star
Number of poles	4
Rated power	3 Hp (~ 2.24 kW)
Line voltage (RMS)	230 V
Line current (RMS)	9 A
Rated speed	1430 rpm
Rated torque	14.96 N m
Rotor construction	Wound rotor with slip rings
Stator resistance	0.55 Ω
Stator inductance	0.068 H
Magnetizing inductance	0.063 H
Rotor resistance	0.72 Ω
Rotor inductance	0.068 H
Moment of inertia	0.05 kg m^2

Figure 2 shows the IM efficiency corresponding to the rated rotor flux reference which is constant regardless of the IM load variation. When the IM load is 80% of the rated load in the period, t = 0.5–2 s, the IM efficiency is high, 73.1%. At t = 2 s, the IM load starts decreasing to 60%, 50%, 40% and 20% of the rated load and the IM efficiency then decreases to 68.8%, 66.2%, 62.2% and 45.1% respectively. When the IM load decreases, the output power decreases and the input power is constant. As a consequence, the IM efficiency decreases. In order to keep high IM efficiency, the input power is required to decrease and this can be achieved by changing the rotor flux reference to its optimal value.

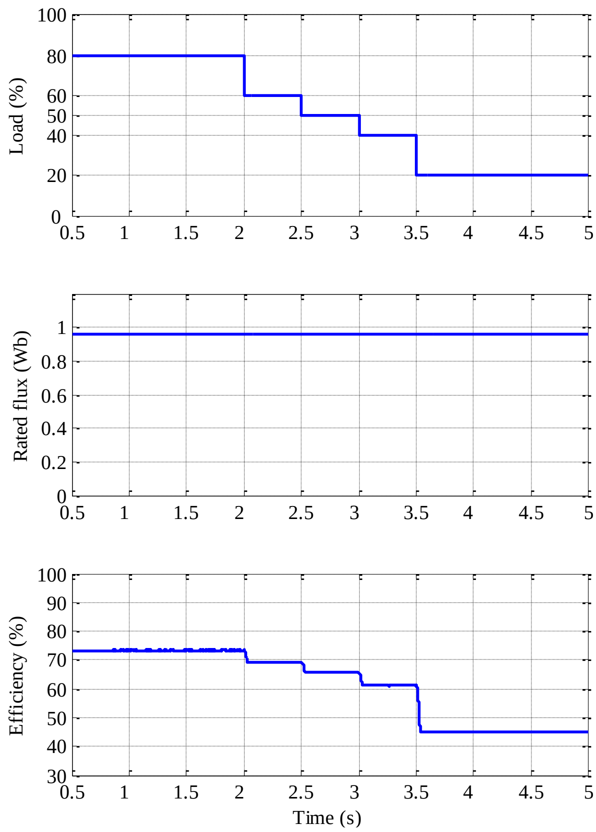

Figure 2. IM efficiency with the rated rotor flux reference.

Figures 3-6 show that the IM always has high efficiency with the optimal IM rotor flux reference obtained by the standard PSO, dynamic PSO and chaos PSO algorithms and the GA. The rotor flux reference alters to adapt to the IM load variations. There is a significant improvement in the IM efficiency, Figures 3-6, which is compared to the IM efficiency using the rated rotor flux reference,

Figure 2, especially at light loads. The IM efficiency is 45.1% at the lightest load whereas it is 72.9%, 81.0%, 83.5% and 79.0% using the optimal rotor flux reference obtained by the standard PSO, dynamic PSO and chaos PSO algorithms and the GA, Table 5.

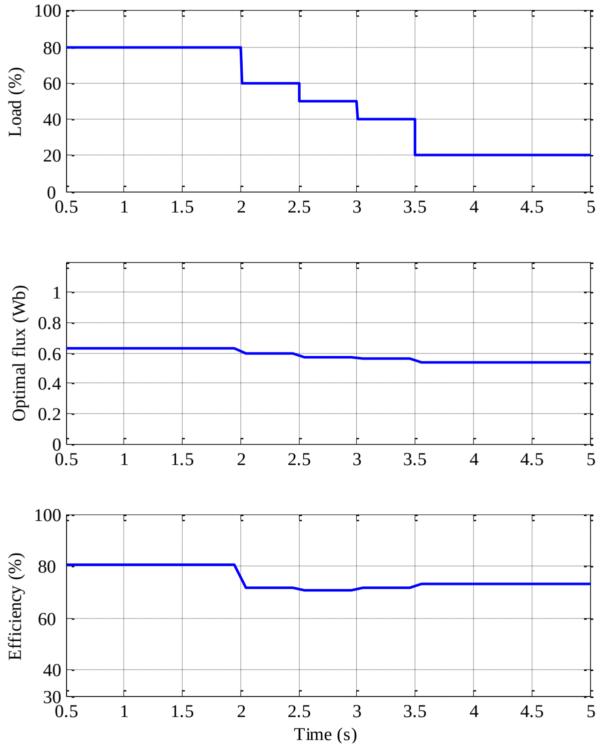

Figure 3. IM efficiency with the optimal rotor flux reference obtained using the standard PSO algorithm.

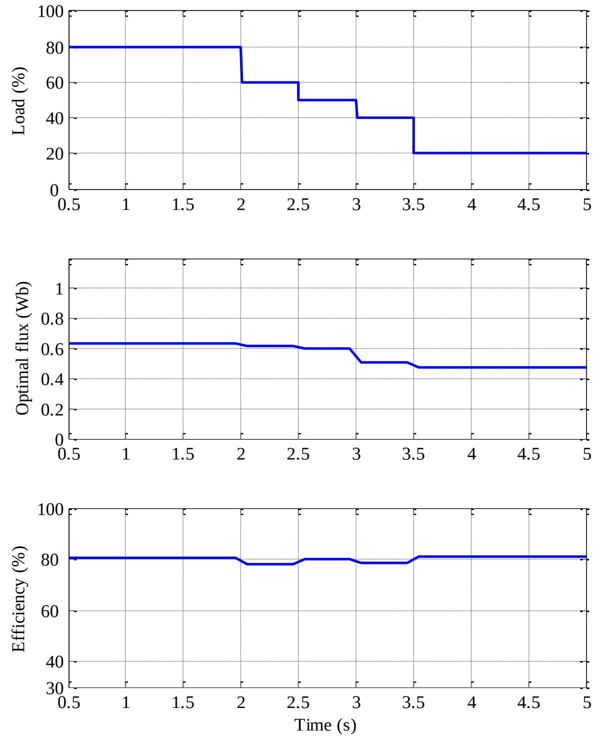

Figure 4. IM efficiency with the optimal rotor flux reference obtained using the dynamic PSO algorithm.

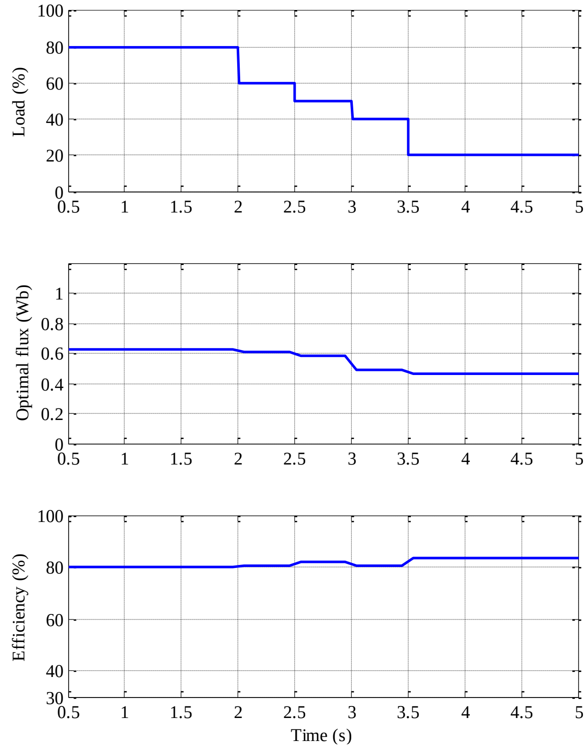

Figure 5. IM efficiency with the optimal rotor flux reference obtained using the chaos PSO algorithm.

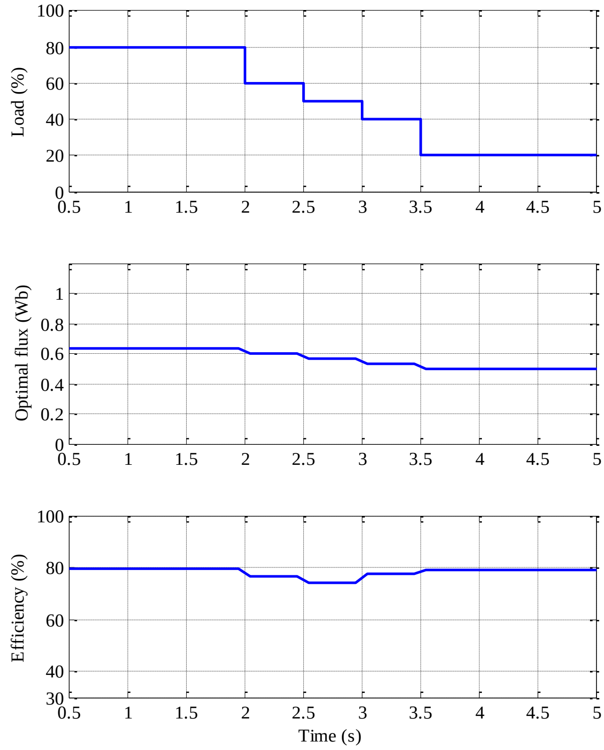

Figure 6. IM efficiency with the optimal rotor flux reference obtained using the GA.

Figures 7–10 are the best fitness of the GA, standard PSO, dynamic PSO and chaos PSO algorithms versus the iteration step number and show the convergence capability of each algorithm.

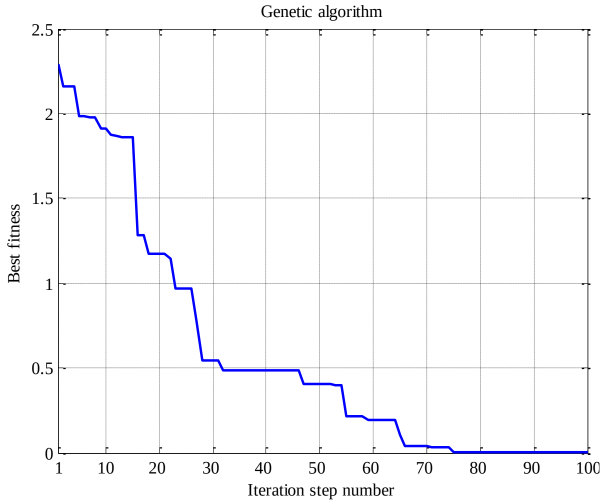

Figure 7. Best fitness versus the iteration step number of the GA.

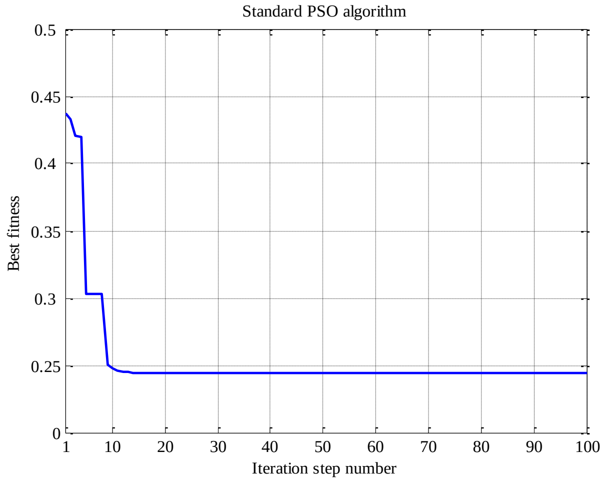

Figure 8. Best fitness versus the iteration step number of the standard PSO algorithm.

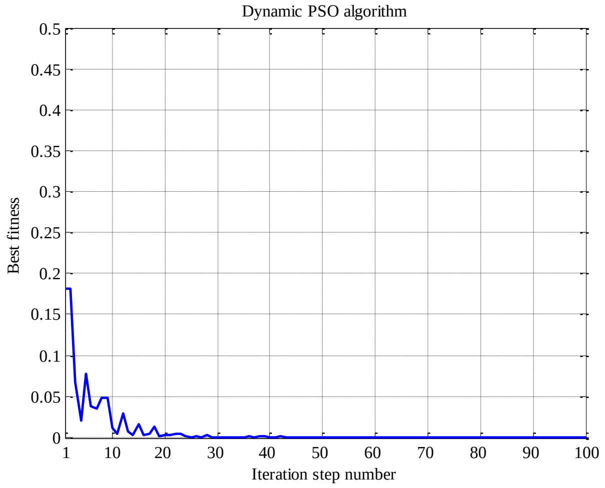

Figure 9. Best fitness versus the iteration step number of the dynamic PSO algorithm.

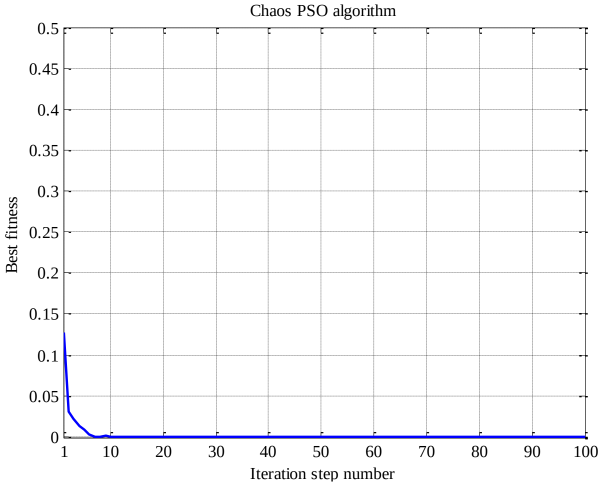

Figure 10. Best fitness versus the iteration step number of the chaos PSO algorithm.

It can be observed that there is a basic difference between the standard PSO and dynamic PSO algorithms from Table 2. The cognitive and social parameters are time-varying variables in the velocity update equation of the dynamic PSO algorithm. This results in a significant improvement in the convergence value of the dynamic PSO algorithm as shown in Figures 8 and 9. Table 3 shows that the convergence value of the standard PSO algorithm is 0.24417 whereas that of the dynamic PSO algorithm is 1.399×10^{-6}.

Table 2. Parameters in the standard PSO, dynamic PSO and chaos PSO algorithms.

Algorithm	Standard PSO	Dynamic PSO	Chaos PSO
Initial particles' positions	Random numbers $\in (0,1)$	Random numbers $\in (0,1)$	Chaotic maps, using (22)
Initial particles' velocities	Random numbers $\in (0,1)$	Random numbers $\in (0,1)$	Chaotic maps, using (23)
Inertia weight, w	w = constant = 0.9	w = constant = 0.9	A chaotic map, using (24)
Acceleration coefficients, c_1 and c_2	$c_1 = c_2$ = constant = 2	Time-varying variables, using (18) and (19)	$c_1 = c_2$ = constant = 2
Independent random sequences, r_1 and r_2	Random numbers $\in (0,1)$	Random numbers $\in (0, 1)$	Chaotic maps, using (25) and (26)

Table 3. The convergence value of algorithms

Algorithm	GA	Standard PSO	Dynamic PSO	Chaos PSO
Convergence value	5.476×10^{-3}	0.24417	1.399×10^{-6}	1.127×10^{-7}

Similarly, several differences also exist between the standard PSO and chaos PSO algorithms in Table 2 such as the initialisation of the particles' positions and velocities using the chaotic map, the chaotic inertia weight and the two chaotic independent random sequences in the velocity update equation of the chaos PSO algorithm. These enhance the solution quality of the algorithm. The convergence value of the chaos PSO algorithm is better than that of the standard PSO algorithm as shown in Figures 8 and 10. Table 3 shows that the convergence value of the standard PSO algorithm is 0.24417 whereas that of the chaos PSO algorithm is 1.127×10^{-7}.

All these features in both the dynamic PSO and chaos PSO algorithms improve the performance as well as avoiding premature convergence in the standard PSO algorithm as illustrated in Figures 8–10. The dynamic PSO and chaos PSO algorithms are therefore better than the standard PSO algorithm.

Additionally, when the standard PSO algorithm is compared with the GA, the standard PSO algorithm converges to the best fitness value faster than the GA in Figures 7 and 8; however this does not mean that the standard PSO algorithm is better than the GA. The standard PSO algorithm became stuck in a local optimum during the search process and resulted in premature convergence. Table 4 shows that the standard PSO algorithm converges at the 14th iteration step whereas the GA converge at the 77th iteration step.

Table 4. The convergence speed of algorithms

Algorithm	GA	Standard PSO	Dynamic PSO	Chaos PSO
Iteration step number	77	14	44	17

Table 5. IM Efficiency with various load variations

Time (s)	IM Load (%)	IM Efficiency (%)				
		Rated Flux	Optimal flux			
			Standard PSO	Dynamic PSO	Chaos PSO	GA
0.5–2	80	73.1	80.4	80.6	80.1	79.6
2–2.5	60	68.8	71.5	78.0	80.7	76.5
2.5–3	50	66.2	70.8	79.9	81.9	74.0
3–3.5	40	62.2	71.6	78.5	80.5	77.7
3.5–5	20	45.1	72.9	81.0	83.5	79.0

When the GA is compared with the dynamic PSO and chaos PSO algorithms, it is observed that the performance of the dynamic PSO and chaos PSO algorithms are better than the GA in terms of both the convergence speed and value in Figures 7, 9 and 10. Table 3 shows that the convergence value of the GA is 5.476×10^{-3} whereas that of the dynamic PSO and chaos PSO algorithms are 1.399×10^{-6} and 1.127×10^{-6} respectively. Furthermore, Table 4 shows that the dynamic PSO and chaos PSO algorithms converge at the 44th and 17th iteration steps respectively whereas the GA converges at the 77th iteration step.

These results show that the both the dynamic PSO and chaos PSO algorithms are better than the GA and standard PSO algorithm in term of both the convergence value and speed for energy efficient control of an IM. This confirms the validity and effectiveness of the dynamic PSO and chaos PSO algorithms in this novel application, Figure 11.

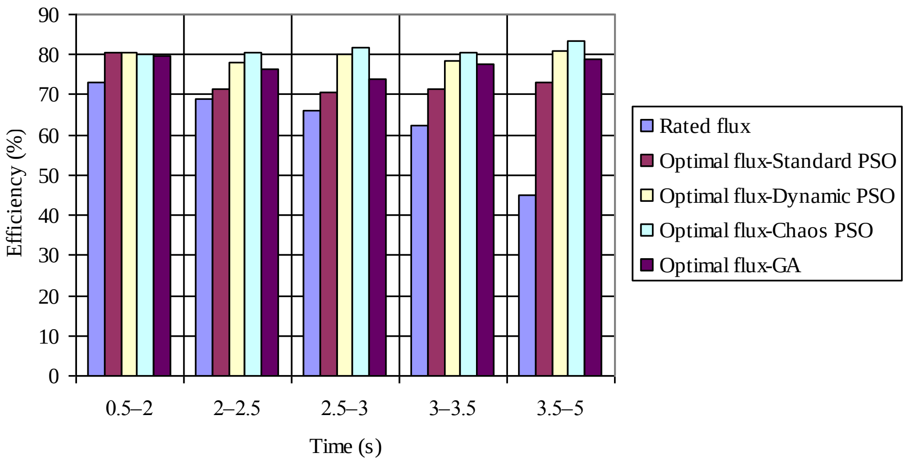

Figure 11. Comparison between IM efficiencies using the rated flux and the optimal fluxes obtained by the standard PSO, dynamic PSO, chaos PSO, and GA.

V. CONCLUSIONS

This paper proposed a novel energy efficient control strategy for the IM using an optimal rotor flux reference obtained by the dynamic PSO and chaos PSO algorithms.

The dynamic PSO algorithm is one of the standard PSO algorithm variants, which modifies the cognitive and social parameters in the velocity update equation of the standard PSO algorithm as linear time-varying parameters. Large cognitive and small social parameters are used in the early part for enhancing the global search capability and then small cognitive and large social parameters are utilized at the end stage to improve the convergence of the algorithm.

The combination of the standard PSO algorithm and the chaotic map is known as the chaos PSO algorithm. The randomness-based parameters of the chaos PSO algorithm are initialized using the logistic map for the initial positions and velocities of the particles, the inertia weight and the two independent random sequences in the velocity update equation. The inertia weight in the chaos PSO algorithm was created for the best balance during the evolution process to produce the best convergence capability and search performance. Furthermore, the algorithm has also been improved because of the diversity in the standard PSO algorithm solution space using two independent chaotic random sequences.

The simulation results show that the IM efficiency is significantly improved, especially for light loads using the optimal rotor flux reference obtained by the standard PSO, dynamic PSO, chaos PSO algorithms and the GA regardless of load variations. It can be realised that the obtained IM efficiency by using the dynamic PSO and chaos PSO algorithms always remained optimal and better than others obtained by using the GA and standard PSO algorithm. Furthermore, the convergence speed and value of the dynamic PSO and chaos PSO algorithms are better than the GA and standard PSO algorithm.

VI. FUTURE WORKS

It can be realised that this proposal has been developed assuming steady-state operation of the IM. Thus, it would be useful to further extend the research for transient conditions.

In this energy efficient control strategy, it is assumed that no measurement noise is present. Thus, it would be useful to examine this effect in future research.

Experimental results for the energy efficient control scheme of the IM would give a valuable confirmation of the simulation results obtained.

REFERENCES

[1]. Hasan, K. M., Zhang, L., and Singh, B., (1997) “Neural network control of induction motor drives for energy efficiency and high dynamic performance”, Proc. 23rd Int. Conf. Industrial Electronics, Control and Instrumentation, IECON ’97, pp488–493.

[2]. Abdin, E. S., Ghoneem, G. A., Diab, H. M. M., and Deraz, S. A., (2003) “Efficiency optimization of a vector-controlled induction motor drive using an artificial neural network”, 29th Annual Conf. IEEE Industrial Electronics Society, IECON’ 03, Virginia, USA, pp2543–2548.

[3]. Perron, M., and Le-Huy, H., (2006) “Full load range neural network efficiency optimization of an induction motor with vector control using discontinuous PWM”, 2006 IEEE Int. Symp. Industrial Electronics, ISIE ’06, Quebec, Canada, pp166–170.

[4]. Pryymak, B., Moreno-Eguilaz, J. M., and Peracaula, J., (2006) “Neural network flux optimization using a model of losses in induction motor drives”, J. Mathematics and Computers in Simulation, Vol. 71, Iss. 4–6, pp290–298.

[5]. Hamid, R. H. A., Amin, A. M. A., Ahmed, R. S., and El-Gammal, A. A. A., (2006) “Optimal operation of induction motors using artificial neural network based on particle swarm optimization (PSO)”, IEEE Int. Conf. Industrial Technology, 2006, ICIT ’2006, pp2408–2413.

[6]. Wang, Z., Xie, S. and Yang, Y., (2009) “A radial basis function neural network based efficiency optimization controller for induction motor with vector control”, 9th Int. Conf. Electronic Measurement & Instruments, 2009, ICEMI ’09, pp866–870.

[7]. Pokier, E., Ghribi, M. and Kaddouri, A., (2001) “Loss minimization control of induction motor drives based on genetic algorithm”, IEEE Int. Electric Machines and Drives Conf., 2001, IEMDC ’2001, Cambridge, MA, USA, pp475–478.

[8]. Rouabah, Z., Zidani, F., and Abdelhadi, B., (2008) “Efficiency optimization of induction motor drive using genetic algorithms”, 4th IET Conf. Power Electronics, Machines and Drives, 2008, PEMD ’2008, York, UK, pp204–208.

[9]. Hamid, R. H. A., Amin, A. M. A., Ahmed, R. S., and El-Gammal, A. A. A., (2006) “New technique for maximum efficiency of induction motors based on particle swarm optimization (PSO)”, 2006 IEEE Int. Symp. Industrial Electronics, ISIE ’06, Quebec, Canada, pp2176–2181.

[10]. Sousa, G. C. D., Bose, B. K., and Cleland, J. G., (1995) “Fuzzy logic based on-line efficiency optimization control of an indirect vector-controlled induction motor drive”, IEEE Trans. Industrial Electronics, Vol. 42, Iss. 2, pp192–198.

[11]. Moreno-Eguilaz, J., Cipolla, M., Peracaula, J., and da Costa Branco, P. J., (1997) “Induction motor optimum flux search algorithm with transient state loss minimization using a fuzzy logic based supervisor”, 28th Annual IEEE Power Electronics Specialists Conf., 1997, PESC ’97, pp1302–1308.

[12]. Zidani, F., Benbouzid, M. E. H., and Diallo, D., (2001) "Loss minimization of a fuzzy controlled induction motor drive", IEEE Int. Electric Machines and Drives Conf., 2001, IEMDC 2001, pp629–633.
[13]. Blanusa, B., Matic, P., and Vukosavic, S. N., (2003) "An improved search based algorithm for efficiency optimization in the induction motor drives", XLVII Konferencija za ETRAN, Herceg Novi, pp1–4.
[14]. Ramesh, L., Chowdhury, S. P., Chowdhury, S., Saha, A. K., and Song, Y. H., (2006) "Efficiency optimization of induction motor using a fuzzy logic based optimum flux search controller", Int. Conf. Power Electronics, Drive and Energy Systems, 2006, PEDES '06, pp1–6.
[15]. De Almeida Souza, D., De Aragao Filho, W. C. P., and Sousa, G. C. D., (2007) "Adaptive fuzzy controller for efficiency optimization of induction motors", IEEE Trans. Industrial Electronics, Vol. 54, Iss. 4, pp2157–2164.
[16]. Cao-Minh Ta, Chakraborty, C., and Hori, Y., (2001) "Efficiency maximization of induction motor drives for electric vehicles based on actual measurement of input power", 27th Annual Conf. IEEE Industrial Electronics Society, IECON '01, Colorado, USA, pp1692–1697.
[17]. Chakraborty, C., Ta, M. C., Uchida, T., and Hori, Y., (2002) "Fast search controllers for efficiency maximization of induction motor drives based on DC link power measurement", Proc. Power Conversion Conf., 2002, PCC '2002, Osaka, pp402–408.
[18]. Vukosavic, S. N., and Levi, E., (2003) "Robust DSP-based efficiency optimization of a variable speed induction motor drive", IEEE Trans. Industrial Electronics, Vol. 50, Iss. 3, pp560–570.
[19]. Liwei, Z., Jun, L., and Xuhui, W., (2006) "Systematic design of fuzzy logic based hybrid on-line minimum input power search control strategy for efficiency optimization of IM", CES/IEEE 5th Int. Power Electronics and Motion Control Conf., 2006, IPEMC '2006, Shanghai, China, pp1–5.
[20]. Sergaki, E. S., and Stavrakakis, G. S., (2008) "Online search based fuzzy optimum efficiency operation in steady and transient states for DC and AC controlled motors", 18th Int. Conf. Electrical Machines, 2008, ICEM 2008, Algarve, Portugal, pp1–7.
[21]. Yanamshetti, R., Bharatkar, S. S., Chatterjee, D., and Ganguli, A. K., (2009) "A hybrid fuzzy based loss minimization technique for fast efficiency optimization for variable speed induction machine", 6th Int. Conf. Electrical Engineering Electronics, Computer, Telecommunications and Information Technology, 2009, ECTI-CON 2009, pp318–321.
[22]. Yanamshetti, R., Bharatkar, S. S., Chatterjee, D., and Ganguli, A. K., (2009) "A dynamic search technique for efficiency optimization for variable speed induction machine", 4th IEEE Conf. Industrial Electronics and Applications, 2009, ICIEA 2009, Xian, China, pp1038–1042.
[23]. Chelliah, T. R., Yadav, J. G., Srivastava, S. P., and Agarwal, P., (2009) "Optimal energy control of induction motor by hybridization of loss model controller based on particle swarm optimization and search controller", World Congress on Nature & Biologically Inspired Computing, 2009, NaBIC 2009, Coimbatore, India, pp1178–1183.
[24]. Bose, B. K., (2002) Modern Power Electronics and AC Drives, Prentice Hall PTR, 2002, pp29–74.
[25]. Dong, G., and Ojo, O., (2006) "Efficiency optimizing control of induction motor using natural variables", IEEE Trans. Industrial Electronics, Vol. 53, Iss. 6, pp1791–1798.
[26]. Kennedy, J., and Eberhart, R., (1995) "Particle swarm optimization", IEEE Int. Conf. Neural Netw., Perth, Australia, pp1942–1948.
[27]. Bergh, F. V. D., (2001) An analysis of particle swarm optimizers, Ph.D. dissertation, Dept. Comput. Sci., Pretoria Univ., Pretoria, South Africa.
[28]. Ratnaweera, A., Halgamuge, S. K., and Watson, H. C., (2004) "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients", IEEE Trans. Evolutionary Computation, Vol. 8, Iss. 3, pp240–255.
[29]. Shi, Y., and Eberhart, R., (1998) "A modified particle swarm optimizer", Proc. 1998 IEEE Int. Conf. Evolutionary Computation, Anchorage, Alaska, USA, pp69–73.
[30]. Alatas, B., Akin, E., and Ozer, A. B., (2009) "Chaos embedded particle swarm optimization algorithms", J. Chaos, Solitons & Fractals, Vol. 40, Iss. 4, pp1715–1734.
[31]. Meng, H. J., Zheng, P., Wu, R. Y., Hao, X. J., and Xie, Z., (2004) "A hybrid particle swarm algorithm with embedded chaotic search", Proc. 2004 IEEE Conf. Cybernetics and Intelligent Systems, Singapore, pp367–371.
[32]. Liu, B., Wang, L., Jin, Y. H., Tang, F., and Huang, D. X., (2005) "Improved particle swarm optimization combined with chaos", J. Chaos, Solitons & Fractals, Vol. 25, Iss. 5, pp1261–1271.
[33]. Feng, Y., Teng, G. F., Wang, A. X., and Yao, Y. M., (2007) "Chaotic inertia weight in particle swarm optimization", Proc. 2nd Int. Conf. Innovative Computing, Information and Control, ICICIC '07, pp475–478.

[34]. Feng, Y., Yao, Y. M., and Wang, A. X., (2007) "Comparing with chaotic inertia weights in particle swarm optimization," Proc. 6th Int. Conf. Machine Learning and Cybernetics, Hong Kong, pp329–333.

Authors

Duy C. Huynh received the B.Sc. and M.Sc. degrees in electrical and electronic engineering from Hochiminh City University of Technology, Hochiminh City, Vietnam, in 2001 and 2005, respectively and Ph.D. degree from Heriot-Watt University, Edinburgh, U.K., in 2010. In 2001, he became a Lecturer at Hochiminh City University of Technology. His research interests include the areas of energy efficient control and parameter estimation methods of induction machines.

Matthew W. Dunnigan received the B.Sc. degree in electrical and electronic engineering (with First-Class Honors) from Glasgow University, Glasgow, U.K., in 1985 and the M.Sc. and Ph.D. degrees from Heriot-Watt University, Edinburgh, U.K., in 1989 and 1994, respectively. He was employed by Ferranti from 1985 to 1988 as a Development Engineer in the design of power supplies and control systems for moving optical assemblies and device temperature stabilization. In 1989, he became a Lecturer at Heriot-Watt University, where he was concerned with the evaluation and reduction of the dynamic coupling between a robotic manipulator and an underwater vehicle. He is currently a Senior Lecturer and his research grants and interests include the areas of hybrid position/force control of an underwater manipulator, coupled control of manipulator-vehicle systems, nonlinear position/speed control and parameter estimation methods in vector control of induction machines, frequency domain self-tuning/adaptive filter control methods for random vibration, and shock testing using electrodynamic actuators.